BBDB User Manual

A phone number and address database program for Emacs
Edition $Revision: 1.51 $, $Date: 2003/08/05 12:11:05 $

by Jamie Zawinski, Matt Simmons and the BBDB Development Team

Copyright © 1991-1994 Jamie Zawinski <jwz@netscape.com>
Copyright © 1997-1999 Matt Simmons <simmonmt@acm.org>
Copyright © 2000-2002 The BBDB Development Team

This is edition $Revision: 1.51 $ of the *BBDB User Manual* for BBDB version 2.34, $Date: 2003/08/05 12:11:05 $.

Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on fnord all copies.

Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, provided that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this manual into another language, under the above conditions for modified versions, except that this permission notice may be stated in a translation approved by the author.

1 BBDB

BBDB is a rolodex-like database program for GNU Emacs. **BBDB** stands for *Insidious Big Brother Database*, and is not, repeat, *not* an obscure reference to the Buck Rogers TV series.

It provides the following features:

- Integration with mail and news readers, with little or no interaction by the user:
  - easy (or automatic) display of the record corresponding to the sender of the current message;
  - automatic creation of records based on the contents of the current message;
  - automatic addition of data to arbitrary fields of the record corresponding to the sender of the current message.
- Listing all records which match a regular expression;
- Listing all records which match a regular expression in a particular field ('`company`' or '`notes,`' for example);

1.1 Installation

This program consists of several groups of files, organized by directory:

lisp	the main program code for the BBDB
tex	TEX support files for bbdb-print, the BBDB printing utility
texinfo	the documentation files for the BBDB
utils	miscellaneous external utility programs
bits	things that have been written as add-ons for **BBDB** but have not yet been merged with the main codebase

1.1.1 General Prerequisites

Various parts of the **BBDB** require extra packages to be available that are not part of the **BBDB** distribution. Please note that with one exception no extra packages (beyond those which ship with both GNU Emacs and XEmacs) are required for the use of **BBDB** core functionality.[1] This one exception applies to XEmacs 20.5 users - the `xemacs-base` package must be installed for the correct operation of the core **BBDB** functionality. The table below lists the requirements of the various portions of the **BBDB**. Please note that the absence of any of the below optional packages will not affect core **BBDB** functionality.

BBDB file	Package needed	GNU 19.34	GNU 20.2	XEmacs >=20.4	XEmacs 20.5

[1] "Core Functionality" is defined as the parts of the **BBDB** used to implement basic record creation ('`M-x bbdb-create`') and searching ('`M-x bbdb`').

`bbdb-ftp`	EFS or Ange-FTP	B	B	B	P
`bbdb-gnus`	Gnus[1]	B	B	B	P
`bbdb-mhe`	MH-E	B	B	B	P
`bbdb-reportmail`	Reportmail			B	P[2]
`bbdb-sc`	Supercite	B		B	P
`bbdb-srv`	`gnuserv` and `itimer`			B	B
`bbdb-vm`	VM[4]			B	P
`bbdb-w3`	`browse-url`	B	B	B	P[3]

NOTES:

1. The old GNUS mail/newsreader should still work. Please keep in mind that you have a relatively recent Emacs (GNU 19.34 or later, XEmacs 19.15 or later), you are probably using Gnus.
2. As of this writing, Reportmail is available as part of the `edit-utils` package.
3. As of this writing, `browse-url` is available as part of the `mail-lib` package.
4. The source release of VM is currently required due to the use of macros from the VM codebase in **BBDB**'s VM integration.

Please also note that the XEmacs package locations are as of this writing. As the XEmacs 20.5 package system is still in development, the locations may change without warning.

1.1.2 Normal User Installation

Configuring the compilation process

First of all, you should run the `configure` script at the toplevel of the distribution. This script will perform a number of checks on your system and generate the '`Makefile`'s accordingly.

The `configure` script also comes with a number of options that lets you customize the compilation process. These options are described below where appropriate.

Byte Compiling the Lisp files

First, you need to byte-compile the appropriate **BBDB** Lisp files. While this is in theory an optional step, it is virtually required in practice due to speed reasons.

In order to byte-compile the lisp files, an Emacs of some sort must be used. By default (at `configure` time), `emacs` and `xemacs` will be tried in that order. If you want to use a special Emacs flavor (or if you want to use `xemacs` at the first place), you should pass the `--with-emacs=PROG` option to `configure`.

In order to successfully compile the **BBDB**, the build process also needs to know the location of the various optional packages. If the directories containing these optional packages are in the default Emacs search path (the `load-path` variable), no other changes need be made for the build process to complete successfully.

If the optional packages are not in the default search path, the build process will not find them unless explicitly told of their location(s). To tell the build process where to find Gnus, MH-E, and/or VM, use the `configure` options `--with-gnus-dir=DIR`, `--with-mhe-dir=DIR`, and/or `--with-vm-dir=DIR` variables respectively. To tell the build process where to find any other package(s), pass the directories containing the lisp files for the package(s) to the `configure` option `--with-other-dirs=DIRS`. If multiple directories are to be added, they should be separated by spaces or colons, and should **not** be quoted. For example, to add the '`/p/local/elisp/footnote`' and '`/p/local/elisp/sc`' directories, call the `configure` script as follows:

```
configure --with-other-dirs=/p/local/elisp/footnote:/p/local/elisp/sc
```

After configuring, run one of the following commands:

`make bbdb`	Build the core, mailer independent, components
`make gnus`	Core components plus `Gnus` support
`make mhe`	Core components plus `MH-E` support
`make rmail`	Core components plus `RMAIL` support
`make vm`	Build the core components with `VM` support
`make all`	Core components plus support for all mailers listed above

You can also combine the above `make` commands. For example, to build the **BBDB** with support for `Gnus` and `VM`, you can do so by typing:

```
make gnus vm
```

Moving the files to their final destination

Lisp files

As stated above, the '`lisp`' subdirectory contains the Emacs Lisp source files for the **BBDB**. Therefore, these files must be in the Emacs `load-path`. There are several ways of doing this, three of which are described below:

- Add the '`lisp`' directory from the source distribution to the `load-path`. This will allow you to run the **BBDB** in-place. This method is recommended for normal users or **BBDB** developers, especially if disk usage is an issue. It is **not** recommended for site-wide installations.
- Link the '`lisp`' directory into your '`site-lisp`' directory. This is for a site-wide installation, but it is subject to the following caveat. If you link the '`lisp`' directory into '`site-lisp`', you will make life more difficult for yourself down the road, as you will not be able to make changes to the source directory (new versions, patches, etc) without having an effect on other users who now depend on it. This directory will automatically be added to the `load-path` when Emacs starts.
- Make a directory whose sole purpose in life is containing the production copies of the **BBDB** source and byte-compiled source files. Either put this directory under '`site-lisp`' (or put it somewhere else and link it into '`site-lisp`'). This directory will automatically be added to the `load-path` when Emacs starts. This is the best of the three listed here, as it allows for a degree of separation between the (possibly changing) source tree and the production code.

TEXfiles

The '`tex`' subdirectory contains the TEXsupport files for bbdb-print, the **BBDB** printing utility (See Section 1.8.2 [bbdb-print], page 33). The three support files, '`bbdb-cols.tex`', '`bbdb-print.tex`', and '`bbdb-print-brief.tex`', must be placed in a directory that is either on the default TEXsearch path or is listed in the `TEXINPUTS` environment variable. If neither of these two options is taken, TEXwill not be able to process the file output by `bbdb-print`.

texinfo files

The '`bbdb.info`' file in this directory contains the documentation for the **BBDB**. This file should either be linked or copied to a directory on the default path for the `info` program or listed in the `INFOPATH` environment variable.

1.1.3 XEmacs Package Installation

NOTE: XEmacs packages are currently supported only under XEmacs versions after and including 20.5. If you are not running such a version of XEmacs, you should install the **BBDB** according to the instructions in Section 1.1.2 [Normal User], page 2.

Configuring / Byte Compiling

The configuration and byte-compilation procedures are the same as in the Normal User installation. See Section 1.1.2 [Normal User], page 2.

Moving the files to their final destination

Support is provided for the automatic installation of the **BBDB** in an XEmacs package directory. The following `configure` options are available for you:

`--with-package-dir=DIR`
: This option sets the root of the XEmacs package directory. By default, '`/usr/local/lib/xemacs/site-packages`' is used.

`--with-symlinks`
: If this option is used, the installation will be done by making symbolic links to the sources instead of copying the files.

`--with-linkpath=PATH`
: Without this option, the installation process uses the output of `pwd` to determine the current directory. If something else should be used, you should provide an alternate name for the BBDB toplevel directory by using `--with-linkpath`. If, for example, `pwd` returns '`/p/local/elisp/bbdb`', but you prefer to use '`/usr/local/elisp/bbdb/...`' for the links, usr this: `configure --with-linkpath=/usr/local/elisp/bbdb`. This option is ignored if `--with-symlinks` is not used.

To perform the (un)installation, use the command `make (un)install-pkg`. This will compile the '`lisp/auto-autoloads.el`' file and will install the appropriate files to the appropriate places. The final installation tree will take the following form:

`$(PACKAGEDIR)/`

`lisp/`

`bbdb/` **BBDB** *lisp source files. This directory contains a copy of all* `.el` *and* `.elc` *files from the '*`lisp`*' source directory, or is a symbolic link to it.*

`info/`

`bbdb.info*`
: **BBDB** *documentation files. These are either copies of the info files from the '*`texinfo`*' source directory, or are symbolic links to them.*

`etc/`

`bbdb/`

`tex/` **BBDB** *support files for bbdb-print. This directory contains a copy of the appropriate files from the '*`tex`*' source directory, or is a symbolic link to it.*

`utils/` **BBDB** *miscellaneous utilities. This directory contains a copy of the appropriate files from the '*`utils`*' source directory, or is a symbolic link to it.*

1.1.4 Initial Configuration

The simplest way to configure the **BBDB** is to include the following forms in your Emacs configuration file:

```
(require 'bbdb)
(bbdb-initialize)
```

Note: The forms above replace the autoloads needed for previous versions of the **BBDB**.

This will set up the **BBDB** for basic querying and record manipulation (the Core Functionality referred to in the Prerequisites section). It will not enable any of the mailreader-, newsreader- or other package-specific **BBDB** features. To enable some or all of these features, the `(bbdb-initialize)` form can be enable as shown below. Alternatively, the features can be enabled manually as described in the following sections.

Modifying `(bbdb-initialize)`

The `bbdb-initialize` function can be used to enable the various package-specific **BBDB** functions. This feature activation is accomplished through the passing of symbols which tell the function which features to activate. These symbols are outlined below and in the Emacs documentation for the `bbdb-initialize`[2]

Initialization symbols for mail and news readers

`gnus` Initialize support for Gnus[3]. If you pass the `gnus` symbol, you should probably also pass the `message` symbol.

`mh-e` Initialize support for the MH-E mail reader.

`rmail` Initialize support for the RMAIL mail reader.

`sendmail` Initialize support for sendmail (*`M-x mail`*)

`vm` Initialize support for the VM mail reader.[4]

Initialization symbols for other packages

`message` Initialize support for Message mode (the mail composition program included with Gnus).

`reportmail`
Initialize support for the Reportmail mail notification package.

`sc` Initialize support for the Supercite message citation package. Additional initialization is required for Supercite to work with the **BBDB**. See Section 1.1.4.8 [Supercite Prep], page 8.

`w3` Initialize support for Web browsers.

Initialization example

To initialize support for Gnus 5.5, Message mode, Supercite, and Web browsers, the following forms would be used:

```
(require 'bbdb)
(bbdb-initialize 'gnus 'message 'sc 'w3)
```

[2] This documentation can be accessed by typing *`C-h f bbdb-initialize RET`*.

[3] If you are using GNUS (not Gnus), and if your GNUS version is 3.14 or older, use the `Gnus` (note the capitalization) symbol.

[4] For the VM initialization to work properly, you must either call `bbdb-initialize` with the `vm` symbol from within your VM initialization file (‘`~/.vm`’), or you must call `bbdb-insinuate-vm` manually from within your VM initialization file.

Manual initialization

If your initialization needs exceed those provided by `bbdb-initialize`, refer to the following sections for a description of the procedures necessary for enabling **BBDB** support for the packages listed above. The procedures described are the same as those carried out by the `bbdb-initialize` function when passed the appropriate symbols. That is, the procedure listed in the RMAIL Prep section below is the same as than executed by `bbdb-initialize` when the `rmail` symbol is passed.

1.1.4.1 Initializing BBDB support for Gnus

To take advantage of the **Gnus** features of the **BBDB**, add one of the following forms to your Emacs configuration file:

For Gnus 3.14 or older:

```
(add-hook 'gnus-Startup-hook 'bbdb-insinuate-gnus)
```

For Gnus 3.15 or newer:

```
(add-hook 'gnus-startup-hook 'bbdb-insinuate-gnus)
```

`bbdb-insinuate-gnus` adds bindings for the default keys to **Gnus** and configures **Gnus** to notify the **BBDB** when new messages are loaded. This notification is required if the **BBDB** is to be able to display **BBDB** entries for messages displayed in **Gnus**.

1.1.4.2 Initializing BBDB support for MH-E

To take advantage of the **MH-E** features of the **BBDB**, add the following form to your Emacs configuration file:

```
(add-hook 'mh-folder-mode-hook 'bbdb-insinuate-mh)
```

`bbdb-insinuate-mh` adds bindings for the default keys to **MH-E** and configures **MH-E** to notify the **BBDB** when new messages are loaded. This notification is required if the **BBDB** is to be able to display **BBDB** entries for messages displayed in **MH-E**.

1.1.4.3 Initializing BBDB support for RMAIL

To take advantage of the **RMAIL** features of the **BBDB**, add the following form to your Emacs configuration file:

```
(add-hook 'rmail-mode-hook 'bbdb-insinuate-rmail)
```

`bbdb-insinuate-rmail` adds bindings for the default keys to **RMAIL** and configures **RMAIL** to notify the **BBDB** when new messages are loaded. This notification is required if the **BBDB** is to be able to display **BBDB** entries for messages displayed in **RMAIL**.

1.1.4.4 Initializing BBDB support for Sendmail

To take advantage of send-mail-mode (the one invoked with `M-x mail`) features of the **BBDB**, add the following form to your Emacs configuration file:

```
(add-hook 'mail-setup-hook 'bbdb-insinuate-sendmail)
```

`bbdb-insinuate-sendmail` enables auto-completion in send-mail-mode.

1.1.4.5 Initializing BBDB support for VM

To take advantage of the **VM** features of the **BBDB**, either add `'vm` to the parameters of the `(bbdb-initialize)` form, or add the following form to your '`~/.vm`' file:

```
(bbdb-insinuate-vm)
```

`bbdb-insinuate-vm` adds bindings for the default keys to **VM** and configures **VM** to notify the **BBDB** when new messages are loaded. This notification is required if the **BBDB** is to be able to display **BBDB** entries for messages displayed in **VM**.

1.1.4.6 Initializing BBDB support for Message mode

To allow the **BBDB** to be used in Message mode, add the following form to your Emacs initialization file:

```
(bbdb-insinuate-message)
```

`bbdb-insinuate-message` adds a binding for *`M-TAB`* to Message mode. This will enable completion of addressees based on **BBDB** records. See Section 1.6.1 [Using Message Mode], page 23 for more details on the operation of Message mode **BBDB** record completion.

1.1.4.7 Initializing BBDB support for Reportmail

To allow the Reportmail package to report information from **BBDB** records for new mail, add the following form to your Emacs initialization file:

```
(bbdb-insinuate-reportmail)
```

`bbdb-insinuate-reportmail` adds to the `display-time-get-field` function to allow access to **BBDB** records during new mail information display. See Section 1.6.2 [Using Reportmail], page 23 for more details on the operation of Reportmail with the **BBDB**.

1.1.4.8 Initializing BBDB support for Supercite

To allow the **BBDB** to assist in the storage of Supercite citations, add the following form to your Emacs initialization file:

```
(bbdb-insinuate-sc)
```

`bbdb-insinuate-sc` adds **BBDB** functions to two Supercite hooks - `sc-post-hook` and `sc-attribs-postselect-hook`. See Section 1.6.3 [Using Supercite], page 23 for more details on the operation of Supercite citation management using the **BBDB**.

Three other Supercite variables must be set/modified to allow the **BBDB** to work with it. These variables are not automatically set as it would be impossible to reliably set them without interfering with other user customizations. The modifications are:

`sc-preferred-attribution-list`
: `"sc-consult"` should be added to the list. An example configuration is:

```
(setq sc-preferred-attribution-list
  '("sc-lastchoice" "x-attribution" "sc-consult"
    "initials" "firstname" "lastname"))
```

`sc-attrib-selection-list`
: The following form should be added to `sc-attrib-selection-list`:

```
'(("sc-from-address"
  (".*" . (bbdb/sc-consult-attr
    (sc-mail-field "sc-from-address"))))))
```

`sc-mail-glom-frame`

The following form should be added to `sc-mail-glom-frame`, to allow the retrieval of the name of a person who is a) in the **BBDB** and b) has only included their net address in the message in question.

```
("^$"  (progn (bbdb/sc-default)
              (list 'abort '(step . 0))))
```

An example configuration is as follows:

```
(setq sc-mail-glom-frame
 '((begin (setq sc-mail-headers-start (point)))
 ("^x-attribution:[ \t]+.*$" (sc-mail-fetch-field t) nil t)
 ("^\\S +:.*$" (sc-mail-fetch-field) nil t)
 ("^$" (progn (bbdb/sc-default)
 (list 'abort '(step . 0))))
 ("^[ \t]+" (sc-mail-append-field))
 (sc-mail-warn-if-non-rfc822-p (sc-mail-error-in-mail-field))
 (end (setq sc-mail-headers-end (point)))))
```

The above is also documented in '`bbdb-sc.el`'. The `bbdb/sc-setup-variables` function has been provided as an example for Supercite variable initialization. Please note that while `bbdb/sc-setup-variables` makes every attempt to safely configure the Supercite variables, it will not always work. Specifically, the variables `sc-attrib-selection-list` and `sc-mail-glom-frame` will not be overridden if they have already been defined.

1.1.4.9 Initializing BBDB support for Web Browsers

To allow URLs to be added to **BBDB** records, add the following form to your Emacs initialization file:

```
(bbdb-insinuate-w3)
```

`bbdb-insinuate-w3` adds the definition of `:` to the W3 keymap.

The other `bbdb-w3` functions, specifically the passing of URLs from **BBDB** records to Web browsers, do not require initialization within the **BBDB**. They do, however, require the configuration of the `browse-url` package so it knows to which Web browser URLs are to be passed. For more details on the operation of `bbdb-w3`, see Section 1.6.4 [Using Web Browsers], page 24.

1.2 The BBDB

This section discusses the basics of the **BBDB** - an overview of the database's layout, and a discussion of the basic **BBDB** manipulation commands.

The database itself lives in a file which is named by the variable `bbdb-file`. If this variable is not set, the database is assumed to be in '`~/.bbdb`'.

1.2.1 Database Fields

The database is organized as a set of records, where each record corresponds to one person or organization. Each record has several fields, and each field is of one of several types. Below, the built-in types are listed, followed by a description of how and why some types can be used more than once in a single record:

Type	Description	Notes
`name`	The name of this person, or none if the record corresponds to an organization.	Single value, single instance.
`company`	The name of this person's organization, or none.	Single value, single instance.
`AKA`	A list of other names for this person.	Multiple values through commas.
`net`	A list of this person's network addresses.	Multiple values through commas.
`address`	A list of postal (physical) addresses for this person.	Multiple values through multiple occurences.
`phone`	A list of telephone numbers for this person.	Multiple values through multiple occurences.
`notes`	Random commentary.	Multiple values through multiple occurences.

The field types listed above can be classified into four categories, as indicated by the comments in the 'Notes' column.

Field types marked as "Single value, single occurrence" may only occur once per record. Each occurrence can only have a single value. For example, there will be only one field of type `name` in a record. It will be named `name`, and will contain a single value (the person's name).

The types marked as "Multiple values through commas" are essentially the same as the single value, single occurrence field types, but with one crucial difference: they can contain multiple values in the form of a comma-separated list. So, for example, while a `name` field with a value of "foo, bar" would be treated as if it contained the single value "foo, bar", a `net` field with the same data would be thought of as having two separate values - "foo" and "bar". As in the single occurrence, single value field types, there will be only one occurrence of each "Multiple values through commas" field type, and the occurrence will have the same name as the type.

The "Multiple values through multiple occurrences" field type is the most flexible of the four listed here. There can be multiple occurrences of each type. This type necessarily does not have the name restriction imposed by the previous two types. For example, there can be multiple fields of the `address` type, none of which have to be named `address`. One could be named `home`, and the other could be named `work`.

Special properties of the `notes` field type: All user-defined fields that don't fit into the other predefined field types (`name`, `company`, `AKA`, `net`, `address`, and `phone`) will be created as a `notes`-type field. In addition, several of the user-defined field names are "special". That is, the **BBDB** treats the values of these "special" fields differently than it does other user-defined fields. The "special" fields are:

`attribution`
: *(Available only when the Supercite-specific* **BBDB** *functions have been enabled)* Used for the storage of Supercite attributions. For initialization details, see Section 1.1.4.8 [Supercite Prep], page 8. For usage details, see Section 1.6.3 [Using Supercite], page 23. The field used can be changed by changing the value of `bbdb/sc-attribution-field`.

`aka`
: Used to store non-primary names associated with a given record.

`face`
: *(XEmacs only)* Used for the storage of image data. This data is to be in the format output by `compface`, and commonly found in `X-Face:` headers in messages. If face support has been compiled into XEmacs, the image contained in the `face` header will be displayed when the record is viewed.

`finger-host`
: Address used in place of the listed net address for fingering the entity indicated by the record. See Section 1.3 [BBDB Mode], page 13. The field used can be changed by changing the value of `bbdb-finger-host-field`.

`gnus-score`
: **Gnus** scoring adjustment for this person. For initialization details, see Section 1.1.4.1 [Gnus Prep], page 7. For usage details, see Section 1.5.1 [Gnus Features], page 19. The field used can be changed by changing the value of `bbdb/gnus-score-field`.

`mail-alias`
: Value used instead of `name` for completion. See Section 1.4.3 [Mail Sending Interfaces], page 18.

`mail-name`
: *(Available only when the Reportmail-specific* **BBDB** *functions have been enabled)* Used for the storage of non-default names to be used in the reporting of new mail by Reportmail. For initialization details, see Section 1.1.4.7 [Reportmail Prep], page 8. For usage details, see Section 1.6.2 [Using Reportmail], page 23.

`mark-char`
: The field containing the character to be used for marking a given poster in the Gnus Summary Buffer. For usage details, see Section 1.5.1.2 [Gnus Summary Buffer], page 20.

`tex-name`
: The value of this field is used in place of the `name` field when printing the database using `bbdb-print`. See Section 1.8.2 [bbdb-print], page 33.

`www`
: This field contains the URL associated with the **BBDB** record. Common uses are with `bbdb-snarf` (see Section 1.8.3 [bbdb-snarf], page 36) and the **BBDB**/Web Browser functionality (for initialization details, see Section 1.1.4.9 [Web Browser Prep], page 9. For usage details, see Section 1.6.4 [Using Web Browsers], page 24).

1.2.2 Basic searching commands

You can list the contents of the database with the command *M-x bbdb*. You will be prompted for a regular expression, and all records which match that regexp in the name, company, network address, or any notes fields will be displayed.

A narrower search may be made by using the commands *bbdb-name*, *bbdb-company*, *bbdb-net*, or *bbdb-notes*. These commands limit their searches to the name, company, email address, and notes fields, respectively. If these commands are given a prefix argument, the listing displayed will be one line per entry; otherwise, the full database entry will be shown on multiple lines.

The functions described above are predefined to certain keys in the `*BBDB*` buffer. See Section 1.3 [BBDB Mode], page 13. for more details.

The `bbdb-notes` command will prompt for the notes field to search (*RET* for all). In this way you can limit your searches to the contents of one particular user-defined notes field. (You can add user-defined fields with the `bbdb-insert-new-field` command; See Section 1.3 [BBDB Mode], page 13.)

1.2.3 Advanced searching commands

The following functions can be used to search for records based on creation and/or modification dates. These functions will match records that have `timestamp` and/or `creation-date` fields (as appropriate. See Section 1.7.3 [Predefined Hooks], page 30. for more information on these fields, which are created by default.

`bbdb-timestamp-older`
: Display all records modified before a given date. If this function is called interactively, it will prompt for a date. If it is being called non-interactively, the date should be provided as a string in '`yyyy-mm-dd`' format.

`bbdb-timestamp-newer`
: Display all records modified after a given date. If this function is called interactively, it will prompt for a date. If it is being called non-interactively, the date should be provided as a string in '`yyyy-mm-dd`' format.

`bbdb-creation-older`
: Display all records created before a given date. If this function is called interactively, it will prompt for a date. If it is being called non-interactively, the date should be provided as a string in '`yyyy-mm-dd`' format.

`bbdb-creation-newer`
: Display all records created after a given date. If this function is called interactively, it will prompt for a date. If it is being called non-interactively, the date should be provided as a string in '`yyyy-mm-dd`' format.

`bbdb-creation-no-change`
: Display all records that have not been changed since creation.

1.2.4 Manual record addition

There are several ways to add new entries to the Insidious Big Brother Database; the most straightforward is to use *M-x bbdb-create*, which will prompt you for all relevant

information. However, the easiest way is to allow them to be added automatically by one of the mail or news-reading interfaces (See Section 1.4 [Interfaces], page 17.).

1.3 BBDB Mode

1.3.1 Functions bound to keys in BBDB Mode

When the '`*BBDB*`' buffer is active (either summoned by one of the commands in the previous section [See Section 1.2 [The BBDB], page 9.] or by your mail or news program), a variety of commands become available for database manipulation. Some of the commands listed below take numeric arguments. These arguments can be generated by entering the number before pressing the key(s) corresponding to the desired command. The output (if any) of the listed commands will be displayed in the '`*BBDB*`' buffer, and can be navigated through using the usual cursor motion commands.

`e` (`bbdb-edit-current-field`) Edit the field on the current line. If the cursor is in the middle of a multi-line field, such as an address or comments section, then the entire field is edited, not just the current line.

`;` (`bbdb-edit-notes`) A shortcut for editing the *notes* field.

`d, C-k` (`bbdb-delete-current-field-or-record`) Delete the field on the current line. If the current line is the first line of a record, the **BBDB** will, after prompting the user, delete the entire record from the database. This may also be applied to multiple records at once by `*`.

`C-o` (`bbdb-insert-new-field`) Inserts a new field into the current record. You are prompted (with completion) for the type of field to insert (**phone**, **address**, **notes**, etc); if the string you type is not a known field type, you will be asked whether to add a new field with the entered name of type `notes`.

If you are inserting a new phone-number field, you can control whether it is a North American or European phone number by providing a prefix argument. A prefix arg of `^U` means it's to be a euronumber, and any other prefix arg means it's to be a a structured North American number. If no prefix argument is supplied, the style used is controlled by the variable `bbdb-north-american-phone-numbers-p`.

`C-x C-t` (`bbdb-transpose-fields`) This is like the `transpose-lines` command, but it is for **BBDB** fields. If the cursor is on a field of a **BBDB** record, that field and the previous field will be transposed.

With non-zero numeric argument *ARG*, the previous field is moved past *ARG* fields. With argument 0, the field indicated by *point* is interchanged with the one indicated by *mark*.

Both fields must be in the same record, and must be of the same basic type (that is, you can use this command to change the order in which phone-number fields are listed, but you can't use it to make an address appear before a phone number; the order of field types is fixed.)

`n, p` (`bbdb-next-record`, `bbdb-prev-record`) Move to the next and previous displayed record, respectively.

t (`bbdb-toggle-records-display-layout`) Toggles the display layout of a record. With a numeric argument of 0, the current record will be made displayed in one line layout; with any other argument, the current record will be shown in multi-line layout.

If `*t` is used instead of simply `t`, then the state of all records will be changed instead of just the one at point. In this case, a numeric argument of 0 means that all records will unconditionally be made one-line layout; any other numeric argument means that all of the records will unconditionally be shown expanded; and no numeric argument means that the records are made to be in the opposite state of the record under point.

T (`bbdb-display-record-completely`) Show all the fields of the current record. The display layout 'full-multi-line' is used for this.

o (`bbdb-omit-record`) Removes the current record from the display, but does not delete it from the database; it merely makes it seem as if the most recent search had not matched this record. With a numeric argument, omit the next N records. With a negative argument, go backwards.

m (`bbdb-send-mail`) Begin composing mail to the person represented by the current record. The first email address is used. Normally, the mail-sending package which is used is determined by which mail-reading package is loaded; that is, if **MH-E** is loaded, then `mh-send` will be used; if **VM** is loaded, then `vm-mail` is used; if **message** is loaded, then it is used; otherwise, `mail` is used. You can override this by setting the variable `bbdb-send-mail-style` to one of the symbols `vm`, `mh`, `message`, or `mail`.

If `*m` is used instead of simply `m`, then mail will be sent to all of the folks listed in the '`*BBDB*`' buffer instead of just the person under point.

This function does not at present use the facility provided by `compose-mail` and `mail-user-agent`. In a future version of the **BBDB**, it will.

s, C-x C-s

(`bbdb-save-db`) Saves the **BBDB** file to disk.

r (`bbdb-refile-record`) Merge the current record into some other record; that is, delete the record under point after copying all of the data within it into some other record. this is useful if you realize that somehow a redundant record has gotten into the database, and you want to merge it with another.

If both records have names and/or companies, you are asked which to use. Phone numbers, addresses, and network addresses are simply concatenated. The first record is the record under the point; the second is prompted for. Completion behavior is as dictated by the variable `bbdb-completion-type`.

M-d (`bbdb-dial`) This command will attempt to dial the phone number currently at point, or if point is at the start of a record, the first phone number in the record. An extension, if present, is disregarded.

The method of dialling is controlled by `bbdb-modem-dial`. If this variable is nil, the **BBDB** will play touchtones corresponding to the number to be dialled. Otherwise, this variable is treated as a modem command string to be prepended to the number prior to feeding it to `bbdb-modem-device`.

The **BBDB** plays touchtones using `bbdb-sound-player` to play the sounds and the elements of `bbdb-sound-files` as the audio to be played. The first ten elements of `bbdb-sound-files` correspond to the touchtones for the digits '`0`' to '`9`', while the eleventh and twelfth elements correspond to '`#`' and '`*`' respectively. The default configuration assumes a Solaris[tm] installation with the demonstration sound files in `/usr/demo/SOUND/sounds`.

The actual number dialled depends on the following variables:

`bbdb-dial-local-prefix-alist`
: This is a list of (SEXPR REPLACEMENT) pairs. SEXPR is evaluated to produce a regular expression which is then applied to the number. If it matches, whatever it matches is replaced by REPLACEMENT. The match and replace is performed using each item in the list that matches, in sequence, so that the output from one item may become input to another. The default value for this variable is to remove `(bbdb-default-area-code)` (i.e. the value of that variable, in parenthesis) from the start of the number to be dialled.

  Note: If this procedure produces a transformed number then no further modifications (such as prefix additions, below) will be made to the number before dialling.

  Using a prefix argument to `bbdb-dial` disables the processing of this variable. The other modifiers, below, are not affected by this.

`bbdb-dial-local-prefix`
: If the number to be dialled starts with a zero, it is deemed to be a local number, and `bbdb-dial-local-prefix` is prepended to it (see note above concerning `bbdb-dial-local-prefix-alist` processing, however).

`bbdb-dial-long-distance-prefix`
: If the number to be dialled starts with a plus sign (`+`), it is deemed to be a long distance number, and `bbdb-dial-long-distance-prefix` is prepended to it (see note above concerning `bbdb-dial-local-prefix-alist` processing, however).

f (`bbdb-finger`) This command fingers the network address of a **BBDB** record. If this command is executed from the '`*BBDB*`' buffer, it fingers the network address of the record which is at point; otherwise, it prompts in the minibuffer (with completion) for a user to finger. With a numeric prefix argument, it fingers the *N*th network address of the current record; with a prefix argument of *^U*, it fingers all of them. The '`*finger*`' buffer is filled asynchronously, meaning that you don't have to wait around for it to finish; but fingering another user before the first finger has finished could have unpredictable results.

If this command is executed from the '`*BBDB*`' buffer, it may be prefixed with `*` (as in **f* instead of simply *f*), meaning to finger all of the users currently listed instead of just the one under point. The numeric prefix argument has the same interpretation.

You can define a special network address to "finger" by defining a field `finger-host`. The name of the field to be fingered can be changed by setting `bbdb-finger-host-field`.

q (`bbdb-bury-buffer`) Hides the '`*BBDB*`' buffer. **Note:** This command does **not** kill the '`*BBDB*`' buffer.

? (`bbdb-help`) This displays a one-line help message in the minibuffer, showing some of the most common bbdb-mode commands.

i (`bbdb-info`) This documentation is displayed. Please note that either '`bbdb`' or '`bbdb.info`' must be installed in one of the info directories known to Emacs for this command to work.

`bbdb-info-file`
: If this documentation is not installed in the standard Info directory, then you should set this variable to the name of the texinfo-formatted version of this file; the `bbdb-info` command will use this file instead.

W (`bbdb-www`) Displays the Web page listed in the `www` field of the current record. See Section 1.6.4 [Using Web Browsers], page 24.

P (`bbdb-print`) Creates a TeX file that contains a pretty-printed version of **BBDB** records. If prefixed by `*`, only the records currently displayed will print. See Section 1.8.2 [bbdb-print], page 33.

h Moves point to another window via the `other-window` function.

c (`bbdb-create`) Create a new database record from information supplied by the user.

C (`bbdb-changed`) Display all records that have been changed since the last time the database was saved.

b (`bbdb`) Begin a new database search. The results of the new search will be displayed in place of the results of the old search.

S a, S c, S o, S n
(`bbdb-net`, `bbdb-company`, `bbdb-notes`, `bbdb-name`)
Begin a new database search. This search will be limited to the net address, company, notes, or name fields, respectively, of database records. See Section 1.2.2 [Basic Searching], page 11. for more details.

*** `bbdb-append-records` will make the next display/search command to append its results to the BBDB buffer instead of replacing its content.

With an prefix arg (C-u) toggle between always append and no append. With an prefix arg that is a positive number append will be enabled for that many times. With any other argument append will be enabled once."

1.3.2 Other database manipulation functions

`bbdb-kill-older`
: If called interactively (or with a single argument - a date in '`yyyy-mm-dd`' format), it will kill all records that were last modified before the given date

as determined by the `timestamp` field. See Section 1.7.3 [Predefined Hooks], page 30. If called non-interactively with a date (in '`yyyy-mm-dd`' format), a comparison function and an action function, the comparison function is applied to the `timestamp` field of all records, and the action function applied to those for whom the comparison function returns true. If '`nil`' is supplied as the comparison function, `string-lessp` is used.

1.4 Interfaces

The **BBDB** interfaces itself with several message-handling packages, but certain parameters control its behavior depending on whether it is being used from within a mail reader or a news reader.

In all of these packages, two new keybindings will be added:

: (`bbdb/`*package*`-show-sender`) Displays the **BBDB** entry corresponding to the author of the current message. If there is none, you will be asked whether to create one. The function called is `bbdb/`*package*`-show-sender`, where *package* is either `gnus`, `mh`, `rmail`, or `vm`, depending on the mail or news program being used when the command is invoked.

; (`bbdb/`*package*`-annotate-sender`) Lets you edit the '`notes`' field of the **BBDB** record corresponding to the sender of the current message. If there is no record for the current author, you will be asked whether to create one. The function called is `bbdb/`*package*`-annotate-sender`, where *package* is either `gnus`, `mh`, `rmail`, or `vm`, depending on the mail or news program being used when the command is invoked.

These keybindings (and several other features) will not be available unless you call the appropriate "insinuation" function; See Section 1.1 [Installation], page 1.

It is possible to configure **BBDB** so that it automatically creates a record when it sees a message from a person who is not in the database. It is also possible to have text automatically added to the notes field of the corresponding record depending on the contents of the message headers. See Section 1.7.2 [Customization Hooks], page 28.

1.4.1 Mail Reading Interfaces

There are BBDB interfaces for the following mail readers:

- **Gnus**, a news- and email- reader written by Lars Magne Ingebrigtsen (based on **GNUS** by Mansanobu Umeda).
- **MH-E**, the Emacs interface to **Mail Handler** (**MH**), from the standard emacs library, but packaged separately from XEmacs since version 20.4.
- **RMAIL**, from the standard emacs library (packaged separately for XEmacs users as of 20.4);
- **View Mail**, by Kyle Jones, version 5.31 or newer;

1.4.2 News Reading Interfaces

There are BBDB interfaces for the following news readers:

- **GNUS**, a newsreader written by Masanobu Umeda.

- **Gnus**, the modern news- and email-reading incarnation of **GNUS**. **Gnus** is written by Lars Magne Ingebrigtsen.

1.4.3 Mail Sending Interfaces

When sending mail, the keystroke *M-TAB* is bound to the function `bbdb-complete-name`. This will take the string that you have typed (from point back to the preceding colon, comma, or the beginning of the line) and will complete that against the contents of the database. What you have typed may be an initial subsequence of a person's full name or network address; if it completes ambiguously, then what you have typed will be replaced with the common portion of the matches. Typing *M-TAB* again will show a list of possible completions. If it completes unambiguously, then an address will be inserted. The variable `bbdb-completion-type` controls whether completion is done on real names, or network addresses, or both. The address inserted is normally of the form `User Name <email-address>`; however, if `User Name` has an address of the form `<user.name@somedomain>`, only the `<email-address>` portion is inserted. This can be overridden by setting `bbdb-dwim-net-address-allow-redundancy` to `t`.

This binding is automatically set by the various insinuation functions documented earlier in this manual. (See Section 1.1.4 [Initial Configuration], page 5.) Briefly, the forms for these functions are:

Gnus `(add-hook 'gnus-Startup-hook 'bbdb-insinuate-gnus)` *for Gnus 3.14 or older*
`(add-hook 'gnus-startup-hook 'bbdb-insinuate-gnus)` *for Gnus 3.15 or newer*

MH-E `(add-hook 'mh-folder-mode-hook 'bbdb-insinuate-mh)`

RMAIL `(add-hook 'rmail-mode-hook 'bbdb-insinuate-rmail)`

sendmail `(add-hook 'mail-setup-hook 'bbdb-insinuate-sendmail)`

VM `(bbdb-insinuate-vm)` *Add to "`~/.vm`' file*

The above forms should be added to your Emacs initialization file, except where otherwise noted.

You can control what "real name" is inserted with the `mail-alias` field: if a record has a `mail-alias` field, then that is used instead of their `name` field.

If the variable `bbdb-completion-display-record` is true (the default) then when you successfully complete an address with *M-TAB*, the corresponding record will be appended to the '`*BBDB*`' buffer. The buffer will not be displayed if it is not already visible, but the record will be displayed there.

When sending mail, you can use the command `bbdb-yank-addresses` to CC the current message to the people currently displayed in the '`*BBDB*`' buffer. This is useful if you are in the midst of sending or replying to a message, and you decide to add some recipients. You can use one of the *M-x bbdb* commands to display the set of people that you want to CC the message to, and then execute this command to add them to the list.

If you are using Jamie Zawinski's '`mail-abbrevs.el`' package, which uses the word-abbrev mechanism for mail aliases, then you can store your mail aliases in the BBDB instead of duplicating the information elsewhere.

If you want a mail alias to be defined for a person, simply add a `mail-alias` field to their record. You may have multiple aliases for the same person; simply separate them with commas.

For convenience there is the function `bbdb-add-or-remove-mail-alias` bound to `a` which adds an alias to one or multiple records when prefixed by a `*`. Called with a prefix argument `C-u` it will remove the given alias.

If more than one person has the same mail-alias, then that alias expands to the addresses of all of those people; in this way you can maintain mailing lists within the BBDB.

When you want to group aliases as in `.mailrc` you may just retained the group aliases in your `.mailrc`.

To actually define the aliases which are stored in the BBDB, call the function `bbdb-define-all-aliases` from your `mail-setup-hook` (or `message-setup-hook` if you use Message mode coming with Gnus). This will search the database, and call `define-mail-alias` to define each of the resulting aliases.

1.5 Reader-specific Features

There are features of the **BBDB** that are available only for specific mail- and news-readers. These features are described below.

The headers which are parsed for email addresses and what records are displayed can be controlled by the following variables: `bbdb-get-addresses-from-headers` controls which headers are parsed for sender addresses when calling the show-sender function of your MUA. `bbdb-get-addresses-to-headers` controls which headers are parsed for recipients addresses when calling the show-all-recipients function of your MUA. When using the pop up feature it will search for the addresses in `bbdb-get-addresses-headers` and display them. By default it will list only the first address, but by setting `bbdb-get-only-first-address-p` to `nil` one will will get records for all addresses.

If there is no MUA specific variable for ignoring certain addresses then those addresses matching `bbdb-user-mail-names` will be ignored.

BBDB adds the bindings `:` for showing all senders and `;` for editing the notes of the sender.

1.5.1 Gnus-specific Features

The **BBDB** can be used to provide score information, or to integrate database information into the **Gnus** Summary buffer or the **GNUS** Subject List.

1.5.1.1 Scoring

The **BBDB** can provide scoring information to **Gnus** in one of two ways.

1. Articles whose authors appear in the **BBDB** and who have `gnus-score` fields will have their scores adjusted by the value contained in that field.
2. Articles whose authors appear in the **BBDB** but who do not have `gnus-score` fields will have their scores adjusted by `bbdb/gnus-score-default`. If `bbdb/gnus-score-default` is `nil`, no score adjustment will be made.

The **BBDB** by default searches the field contained in `bbdb/gnus-score-field` for score values. To have the **BBDB** use a different field, change the value of this variable.

To enable **BBDB**-assisted scoring, add the `bbdb/gnus-score` function to `gnus-score-find-score-files-function`. Assuming that you want to preserve the default value of this variable, use a form similar to the following:

```
(setq gnus-score-find-score-files-function
      '(gnus-score-find-bnews bbdb/gnus-score))
```

Note: The default value in **Gnus** 5.5 is `gnus-score-find-bnews`. Check your configuration before using the above code, as your values may be different.

1.5.1.2 Gnus Summary Buffer Enhancements

Gnus can use the **BBDB** to do one of two things:

- Mark authors in the Summary Buffer who have records in the **BBDB** with a user-defined mark character. See Marking Posters, below.
- For authors in the Summary Buffer who also have records in the **BBDB**, replace their name as listed in the Summary Buffer with their name as stored in the **BBDB**. See Using Names from the **BBDB**, below.

Marking Posters

Authors with records in the **BBDB** can be marked either with a user-defined mark character, or with a default one. The marking is enabled by the use of a Gnus user format code, as determined by `bbdb/gnus-summary-in-bbdb-format-letter`. This variable, which defaults to '`b`', is used to create a format code which is intended for use in `gnus-summary-line-format`. The format code is created by concatenating '`%u`' with the value of `bbdb/gnus-summary-in-bbdb-format-letter`. In the default case this results in the creation of the format code '`%ub`'.

Posts are marked as follows: If the record for the poster has the field indicated in `bbdb-message-marker-field` (the default is `mark-char`), the value of that field is used as the mark character.[5] If no such field is present, the value of `bbdb/gnus-summary-known-poster-mark` will be used instead. If the author is not in the **BBDB**, a space will be used as the mark character.

Using Names from the BBDB

The names reported for authors of posts in the Summary buffer can be altered to conform to the values present in their respective **BBDB** records (if any). This rewriting is enabled by the use of a Gnus user format code, as determined by `bbdb/gnus-summary-user-format-letter`. This variable, which defaults to '`B`', is used to create a format code which is intended for use in `gnus-summary-line-format`. The format code is created by concatenating '`%u`' with the value of `bbdb/gnus-summary-user-format-letter`. In the default case this results in the creation of the format code '`%uB`'. This format code is intended to **replace** the format code previously used in the Summary buffer format line to indicate the author and/or net address (usually '`%a`', '`%n`', and/or '`$N`').

[5] While it is possible to put a multi-character mark in `bbdb-message-marker-field` and/or in `bbdb/gnus-summary-known-poster-mark`, the resulting summary buffer will be misaligned as a result. This misalignment will result from fact that at this time the character used to indicate posts whose authors are not in the **BBDB** is always a single character, and cannot be changed.

The effects of this format code are in two independent parts - the marking of known posters, and the rewriting of posters names. The first, the marking of posters, occurs only when `bbdb/gnus-summary-mark-known-posters` is `t` (the default) and the posters have entries in the **BBDB**. When this variable is true, the marking occurs as described in the previous section, Marking Posters, above.

The poster name rewriting is done for all posters - not just for those with records in the **BBDB**. That said, rewriting rules for posters in the **BBDB** are more flexible than for those not listed. The rewriting is governed by two variables, as described below.

`bbdb/gnus-summary-prefer-real-names` can have one of three values - '`t`', '`bbdb`', or `nil`. In general, this variable governs the preference between net addresses and names. If it is '`t`', the name (if any) will be used. If '`nil`', the net address will be used. The third value, '`bbdb`', can be used as a method for distinguishing between authors with records in the **BBDB** and those without. If the variable is set to '`bbdb`', the name from the **BBDB** record will be used if the author has a record in the **BBDB**. If the author is not in the **BBDB**, the net address from the message will be printed. This variable makes little sense if `bbdb/gnus-summary-prefer-bbdb-data` is '`nil`', as no names will be printed in the Summary buffer in this case - only net addresses.

`bbdb/gnus-summary-prefer-bbdb-data` is used to (dis)allow use of the **BBDB** for author data retrieval. If it is '`t`', data from the **BBDB** will be used if available. If it is '`nil`', data from the **BBDB** will not be used.

In the following examples, assume the following:

1. Message: `From: Jamie <jwz@netscape.com>`
   BBDB: No record
2. Message: `From: Matt <simmonmt@acm.org>`
   BBDB: Name: '`Matthew`', Net: '`simmonmt@purdue.edu`'

`bbdb/gnus-summary-prefer-bbdb-data`			
	`t`	`t`	`nil`
`bbdb/gnus-summary-prefer-real-names`			
	`t`	`bbdb`	`t`
Printed in Summary buffer for			
Case 1	Jamie	jwz@netscape.com	Jamie
Case 2	Matthew	Matthew	Matt

1.5.1.3 GNUS Summary Buffer Enhancements

This section is remarkably terse, as I don't have a copy of **GNUS**. *If anybody can provide more descriptive information, please let me know.*

```
(autoload 'bbdb/gnus-lines-and-from "bbdb-gnus")
(setq gnus-optional-headers 'bbdb/gnus-lines-and-from)
```

`bbdb/gnus-mark-known-posters`
: If `t` (the default), then the **GNUS** subject list will contain an indication of those messages posted by people who have entries in the Insidious Big Brother Database (they will be marked with an asterisk.)

  You can change the character used to mark records on a record-by-record basis by adding a `mark-char` property to the record, whose value is be the string to display (preferably one character.)

`bbdb/gnus-header-prefer-real-names`
: Default: `nil`. if `t`, then the **GNUS** subject list will display real names instead of network addresses.

`bbdb/gnus-header-show-bbdb-names`
: Default: `t`. If both this variable and the `bbdb/gnus-header-prefer-real-names` variable are true, then for news messages from people who are in your database, the name displayed will be the primary name from the database, rather than the one from the '`From:`' line of the message. This doesn't affect the names of people who aren't in the database, of course.

`bbdb/gnus-lines-and-from-length`
: Default: 18. The number of characters used to display '`From:`' info in **GNUS**, if you have set `gnus-optional-headers` to `bbdb/gnus-lines-and-from`.

1.5.2 VM-specific features

The **BBDB** can be used to integrate database information into the message summary.

1.5.2.1 VM Message Summary Enhancements

VM users can cause their summary buffer to display the name of the message sender according to **BBDB** data, instead of according to the contents of the current message's headers. In VM 5.40 or later, use the summary format control `%UB"` instead of `"%F"`, and the current record name will be shown there if available. If no entry is found it behaves like `"%F"`. See the documentation for `vm-summary-format` for more details. Warning, this may significantly slow down summary generation for large folders.

1.5.2.2 VM configuration of what records the BBDB buffer shows

Email addresses which match `vm-summary-uninteresting-senders` are ignored.

The records in the **BBDB** buffer are listed in the same order as found.

1.5.2.3 VM automatic setup of `vm-set-auto-folder-alist`

VM users can setup the `vm-set-auto-folder-alist` automatically by calling `bbdb/vm-set-auto-folder-alist`. This adds for each **BBDB** record containing a `bbdb/vm-set-auto-folder-alist-field` an entry to `vm-set-auto-folder-alist`.

The record field can contain a string which is used as folder name or if it starts with a `'` it is treated as lisp expression returning a folder name.

1.5.2.4 VM automatic adding of labels

`bbdb/vm-auto-add-label-list` is a List used by `bbdb/vm-auto-add-label` to automatically label messages. Each element in the list is either a string or a list of two strings. If

a single string then it is used as both the field value to check for and the label to apply to the message. If a list of two strings, the first is the field value to search for and the second is the label to apply.

`bbdb/vm-auto-add-label-field` `bbdb-define-all-aliases-field` is the field used by `bbdb/vm-auto-add-label` to automatically label messages. Value is either a single symbol or a list of symbols of bbdb fields that `bbdb/vm-auto-add-label` uses to check for labels to apply to messages. Defaults to `bbdb-define-all-aliases-field` which is typically `mail-alias`.

`bbdb/vm-auto-add-label` automatically adds labels to messages based on the `bbdb/vm-auto-add-label-field` `bbdb-define-all-aliases-field` field. Add this to `bbdb-notice-hook` and if using VM each message that bbdb notices will be checked. If the sender has a value in the `bbdb/vm-auto-add-label-field` in their BBDB record that matches a value in `bbdb/vm-auto-add-label-list` then a VM label will be added to the message.

This works great when 'bbdb-user-mail-names' is set. As a result mail that you send to people (and copy yourself on) is labeled as well.

1.6 Using the BBDB with other packages

The **BBDB** adds functionality to several packages. The following sections detail these augmentations.

1.6.1 Using the BBDB with Message Mode

At this time, the only feature the **BBDB** adds to Message mode is the binding to *M-TAB* which allows for **BBDB** record completion.

1.6.2 Using the BBDB with Reportmail

The **BBDB** can modify the '`reportmail.el`' package to use information from **BBDB** records when identifying the senders or recipients of e-mail messages.

In normal operation, Reportmail displays the name and net address sender and recipient of incoming messages. The **BBDB** can be configured to intercept and rewrite this information before it appears in the Emacs mode-line. It first attempts to rewrite the sender and/or recipient information by substituting those addresses with information from the **BBDB**. Replacement information is first sought from the `mail-name` field in the respective **BBDB** records. If no such field is found, the `name` field is returned. If no **BBDB** record is found, no rewriting is performed.

The **BBDB**-Reportmail augmentation is accomplished through the advising of the Reportmail `display-time-get-field` function in order to do a-posteriori modification of the returned value. The augmentation uses the `bbdb/reportmail-alternate-full-name` function to retrieve data from the **BBDB** for use in rewriting.

1.6.3 Using the BBDB with Supercite

The **BBDB** can be used with Supercite to store attributions with **BBDB** records. Normally, when a non-default attribution is entered for a given message, the entered attribution is used for that message, and is then discarded. When the **BBDB**-Supercite augmentation is enabled, the non-default attribution will be added to the record (if any) for the entity being

cited. This poor explanation sounds complicated, but it's not. If a message from '`Jamie Zawinski <jwz@netscape.com>`' is being replied to, Supercite will, by default, suggest the citation '`Jamie`'. If the non-default citation '`jwz`' is entered, Supercite can save it with the **BBDB** record for '`Jamie Zawinski`' in the `attribution` field.

The field used can be changed by changing the value of `bbdb/sc-attribution-field`.

1.6.4 Using the BBDB with Web Browsers

The **BBDB**/Web Browser integration is in two parts, one which is automatically enabled, and one which must be manually enabled (see Section 1.1.4.9 [Web Browser Prep], page 9). The first feature added is the ability to display the URL associated with a given record in a Web Browser. The second is the ability to add URLs to **BBDB** records from within W3, the Emacs Web Browser.

Pressing `W` in the `*BBDB*` buffer while the cursor is positioned over a record with a `www` field will cause the first URL in the field to be loaded in a Web Browser. This functionality uses `browse-url` to display URLs - see the documentation for `browse-url` for information on selecting the browser to be used.

If W3 is used, and if the **BBDB**/W3 functionality has been enabled as described in Section 1.1.4.9 [Web Browser Prep], page 9, pressing the `:` key will add the URL currently being displayed in W3 to a user-specified **BBDB** record.

1.7 Options

There are many variables which control the behavior of the Insidious Big Brother Database, and there are many hook-variables which can be used to modify its behavior in more complex ways. Several pieces of functionality are included which use the hooks in this way.

1.7.1 Customization Parameters

`bbdb-file`
: The name of the file which contains your personal database. Default: '`~/.bbdb`'.

`bbdb-default-area-code`
: The default area code to use when prompting for a new phone number. Default: 415. This must be a number, not a string.

`bbdb-north-american-phone-numbers-p`
: Whether syntax-checking of phone numbers should be enforced. Default: `t`. This only works for Bell-system phone numbers. If this is true, then you can't enter invalid phone numbers, and all phone numbers are pretty-printed in the same way. European phone numbers don't have as strict a syntax, however, so this is a harder problem for them (on which I am punting).

  You can have both styles of phone number in your database by providing a prefix argument to the `bbdb-insert-new-field` command.

`bbdb-check-zip-codes-p`
: Whether syntax-checking of zip codes should be enforced. Default: `t`. If this is true, you can't enter invalid zip codes. A zip code is valid if it matches one of the regular expressions in the variable `bbdb-legal-zip-codes`.

`bbdb-address-formatting-alist`
: Controls the display of addresses in the buffer. Each entry in this list consists of an identifying function and a formatting function. The identifying function must accept an address and return `t` if the associated formatting function is to be used. The formatting function must insert the formatted address in the current buffer. Identifying functions usually base their decision on the zip code format or on the country name. The default entries will format an address using continental style if the zip code matches `bbdb-continental-zip-regexp`. If the zip code does not match, addresses are formatted in US style.

`bbdb-continental-zip-regexp`
: Decides whether an address should be formatted using US or European style. If the zip code of an address matches the regular expression, the European style is used. This works only if the expression `(bbdb-address-is-continental . bbdb-format-address-continental)` is part of `bbdb-address-formatting-alist`.

`bbdb-electric-p`
: Whether bbdb mode should be *"electric"* like `electric-buffer-list`. Default: `t`. What this means is that the BBDB buffer which pops up when you use it can be disposed of by pressing the space bar, at which point your window configuration will be restored to what it was before you invoked the db list. (The `bbdb-mode` commands still work as well.)

  There are some problems with electric modes; for example, keyboard macros and incremental search don't work. (This is not a bug in BBDB, but in '`electric.el`'.)

`bbdb-case-fold-search`
: Default: the same as `case-fold-search`. `case-fold-search` is bound to this by *`M-x bbdb`* and related commands. This variable lets the case-sensitivity of *`^S`* and of the bbdb searching commands be different.

`bbdb/mail-auto-create-p`
: If this is `t` (the default), then **VM**, **MH**, and **RMAIL** will automatically create new bbdb records for people you receive mail from. If this variable is a function name or lambda expression, then it is called with no arguments to decide whether an entry should be automatically created. You can use this to, for example, not create records for messages which have reached you through a particular mailing list, or to only create records automatically if the mail has a particular subject. See the variables `bbdb-ignore-most-messages-alist` and `bbdb-ignore-some-messages-alist` (See Section 1.7.3 [Predefined Hooks], page 30.)

`bbdb/news-auto-create-p`
: If this is `t` (default: `nil`), then **GNUS** will automatically create new **BBDB** records for people you read messages by. If this is a function name or lambda expression, then it is called with no arguments to decide whether an entry should be automatically created. You can use this to, for example, create or not create messages which have a particular subject. See the variable `bbdb-auto-notes-alist` (See Section 1.7.3 [Predefined Hooks], page 30.).

If you want to autocreate messages based on the current newsgroup, it's probably a better idea to set this variable to `t` or `nil` from your `gnus-select-group-hook` instead.

To automatically remember users in certain groups, you can do something like

```
(setq gnus-select-group-hook
'(lambda ()
(setq bbdb/news-auto-create-p
(or (string= "some.news.group" gnus-newsgroup-name)
(string= "other.news.group" gnus-newsgroup-name)))))
```

`bbdb-quiet-about-name-mismatches`

If this is false (the default), then **BBDB** will prompt you when it notices a name change, that is, when the "real name" in a message doesn't correspond to a record already in the database with the same network address. As in, *"John Smith <jqs@frob.com>"* versus *"John Q. Smith <jqs@frob.com>"*. If this is true, then you will not be asked if you want to change it (and it will not be changed.) If a number then it is the number of seconds to sit-for while displaying the name mismatch.

`bbdb-use-alternate-names`

If this is false, then the **BBDB** will not use the **AKA** field. Otherwise (the default) then the mail and news interfaces will ask you if you want to add an alternate name when a name-change is noticed, and will ask you whether the new name should be made the primary one. Note that if `bbdb-quiet-about-name-mismatches` is true, you will not be asked any questions about alternate names.

`bbdb-readonly-p`

If this is true (default: `nil`), then nothing will attempt to change the database implicitly, and you will be prevented from doing it explicitly. If you have more than one emacs running at the same time, you might want to arrange for this to be set to `t` in all but one of them.

`bbdb-auto-revert-p`

If this variable is true (default: `nil`) and the **BBDB** file is noticed to have changed on disk, it will be automatically reverted without prompting you first. Otherwise you will be asked. (But if the file has changed and you have made changes in memory as well, you will always be asked.)

`bbdb-notice-auto-save-file`

If this is true (default: `nil`), then the **BBDB** will notice when its auto-save file is newer than the file is was read from, and will offer to revert.

`bbdb-use-pop-up`

If true (the default), display a continuously-updating **BBDB** window while in **VM**, **MH**, **RMAIL**, or **GNUS**. Each time a new message is selected, the record corresponding to that message's sender will be displayed in another window. The buffer in this other window will be in bbdb-mode, and all corresponding commands will be available.

This buffer will be positioned on the screen by finding the tallest of the windows present, and splitting it such that the bottom `bbdb-pop-up-target-lines` lines of the window display the '`*BBDB*`' buffer. With the default configurations of **VM**, **MH**, **RMAIL**, and **GNUS**, this means that the bbdb-list buffer will be just below the message-body buffer.

If this is the symbol `horiz`, then the **BBDB** window will be stacked horizontally instead of vertically, if there is room to do that tastefully.

`bbdb-pop-up-target-lines`
: Desired number of lines in a **VM/MH/RMAIL/GNUS** pop-up **BBDB** window, default 5.

`bbdb-completion-type`
: Controls the behavior of the `bbdb-complete-name` command. If `nil` (the default), completion is done across the set of all full-names and user-ids in the database; if the symbol `name`, completion is done on real-names only; if the symbol `net`, completion is done on network addresses only; if it is `primary`, then completion is done only across the set of primary network addresses (the first address in the list of addresses for a given user). If it is `primary-or-name`, completion is done across primaries and real names.

`bbdb-expand-mail-aliases`
: If non-nil, expand mail aliases in 'bbdb-complete-name'.

`bbdb-complete-name-allow-cycling`
: Whether to allow cycling of email addresses when calling 'bbdb-complete-name' on a completed address in a composition buffer."

`bbdb-complete-name-full-completion`
: Show full expanded completion rather than partial matches. If t then do it always, if a number then just is the number of completions for a specific match is below that number.

`bbdb-user-mail-names`
: A regular expression identifying the addresses that belong to you. If a message from an address matching this is seen, the **BBDB** record for the '`To:`' line will be shown instead of the one for the '`From:`' line. If this is `nil`, it will default to the value of `(user-login-name)`.

`bbdb-always-add-addresses`
: If this is `t`, then whenever the Insidious Big Brother Database notices a new email address corresponding to a person who is in the database, it will add it to the database. If this is `nil` (the default), then whenever a new network address is noticed for a person in the database, you will be asked whether to add the address. If this is the symbol `never` (really if it is not `t` and not `nil`) then new network addresses will never be automatically added.

`bbdb-new-nets-always-primary`
: If this is `t`, then when the Insidious Big Brother Database adds a new address to a record, it will always add it to the front of the list of addresses, making it the primary address. If this is `nil` (the default), then you will be asked. If

this is the symbol `never` (really if it is not `t` and not `nil`) then new network addresses will always be added to the end of the list.

`bbdb-canonicalize-redundant-nets-p`
: If this is non-`nil`, redundant network addresses will be ignored. If a record has an address of the form `foo@baz.com`, setting this to `t` will cause subsequently-noticed addresses like `foo@bar.baz.com` to be ignored (since we already have a more general form of that address.) This is similar in function to one of the possible uses of the variable `bbdb-canonicalize-net-hook` but is somewhat more automatic. (This can't quite be implemented in terms of the canonicalize-net-hook because it needs access to the database to determine whether an address is redundant, and the canonicalize-net-hook is purely a textual manipulation which is performed before any database access.)

`bbdb-message-caching-enabled`
: Whether caching of the message->bbdb-record association should be used for the interfaces which support it (**VM**, **MH**, and **RMAIL**). This can speed things up a lot. One implication of this variable being true (the default) is that the `bbdb-notice-hook` will not be called each time a message is selected, but only the first time. Likewise, if selecting a message would generate a question (whether to add an address, change the name, etc) you will only be asked that question the very first time the message is selected.

`bbdb-offer-save`
: If `t` (the default), then certain actions will cause the **BBDB** to ask you whether you wish to save the database. If `nil`, then the offer to save will never be made. If not `t` and not `nil`, then any time it would ask you, it will just save it without asking.

1.7.2 Customization Hooks

All of the hooks variables described below may be set to a symbol or lambda expression, which will be funcalled; or may be set to a list of symbols or lambda expressions, each of which will be funcalled in turn. Almost all hooks in Emacs work this way. But notice that some of the hooks described below are called with arguments.

`bbdb-list-hook`
: Hook or hooks invoked after the bbdb-list-buffer is filled in. Invoked with no arguments.

`bbdb-create-hook`
: Hook or hooks invoked each time a new bbdb-record is created. Invoked with one argument, the new record. This is called *before* the record is added to the database. Note that `bbdb-change-hook` will be called as well.

`bbdb-change-hook`
: Hook or hooks invoked each time a bbdb-record is altered. Invoked with one argument, the record. This is called *before* the database buffer is modified. Note that if a new bbdb record is created, both this hook and `bbdb-create-hook` will be called.

`bbdb-mode-hook`
: Hook or hooks invoked when the '`*BBDB*`' buffer is created.

`bbdb-notice-hook`
: Hook or hooks invoked each time a bbdb-record is "noticed," that is, each time it is displayed by the news or mail interfaces. Invoked with one argument, the new record. The record need not have been modified for this to be called - use `bbdb-change-hook` for that. You can use this to, for example, add something to the notes field based on the subject of the current message. It is up to your hook to determine whether it is running in **GNUS**, **VM**, **MH**, or **RMAIL**, and to act appropriately.

  Also note that `bbdb-change-hook` will *not* be called as a result of any modifications you may make to the record inside this hook.

  Beware that if the variable `bbdb-message-caching-enabled` is true (a good idea) then when you are using **VM**, **MH**, or **RMAIL**, this hook will be called only the first time that message is selected. (The **GNUS** interface does not use caching.) When debugging the value of this hook, it is a good idea to set caching-enabled to `nil`.

`bbdb-after-read-db-hook`
: Hook or hooks invoked (with no arguments) just after the Insidious Big Brother Database is read in. Note that this can be called more than once if the **BBDB** is reverted. One possible use for this is to rename the '`.bbdb`' buffer; for example

  ```
  (setq bbdb-after-read-db-hook '(lambda () (rename-buffer " bbdb")))
  ```

  will cause the buffer visiting the `bbdb-file` to be called `" bbdb"`. The leading space in its name will prevent it from showing up in the buffer list.

`bbdb-load-hook`
: Hook or hooks invoked (with no arguments) when the Insidious Big Brother Database code is first loaded. WARNING: Slow functions should not be put on this hook, as the **BBDB** code will, if not loaded before, be loaded during the first use of **BBDB**-related Customization functions. Slow functions should be put on `bbdb-initialize-hook`.

`bbdb-initialize-hook`
: Hook or hooks invoked (with no arguments) when the `bbdb-initialize` function is called.

`bbdb-canonicalize-net-hook`
: If this is non-`nil`, it should be a function of one argument: a network address string. (Note that, unlike the other hook-variables described above, this may not be a list of functions.) Whenever the Insidious Big Brother Database "notices" a message, the corresponding network address will be passed to this function first, as a kind of "filter" to do whatever transformations upon it you like before it is compared against or added to the database. For example: it is the case that `CS.CMU.EDU` is a valid return address for all mail originating at a machine in the `.CS.CMU.EDU` domain. So, if you wanted all such addresses to be canonically hashed as `user@CS.CMU.EDU`, instead of as `user@somehost.CS.CMU.EDU`, you might set this variable to a function like this:

```
(setq bbdb-canonicalize-net-hook
      '(lambda (addr)
         (cond ((string-match
                  "\\`\\([^@]+@\\).*\\.\\(CS\\.CMU\\.EDU\\)\\'"
                  addr)
                (concat (substring addr
                          (match-beginning 1) (match-end 1))
                        (substring addr
                          (match-beginning 2) (match-end 2))))
               (t addr))))
```

You could also use this function to rewrite UUCP-style addresses into domain-style addresses, or any number of other things.

This function will be called repeatedly until it returns a value EQ to the value passed in. So multiple rewrite rules might apply to a single address.

There is an example of the use of this variable in the file 'bbdb-hooks.el': the function sample-bbdb-canonicalize-net-hook.

The bbdb-canonicalize-net-hook is powerful in that it allows arbitrary rewriting of addresses, however, in many cases that is overkill. The function bbdb-delete-redundant-nets can be used as a value of bbdb-change-hook to cause network addresses which appear to be "redundant" to be deleted each time a modification is made to a record.

This works as follows: suppose one gets mail from user@foo.bar.com, and then later gets mail from user@bar.com. At this point, one can generally delete the user@foo.bar.com address, since the user@bar.com address is more general. (See also the variable 'bbdb-canonicalize-redundant-nets-p', which has the effect of ignoring subsequent addresses from user@quux.bar.com if the address user@bar.com is already known.)

1.7.3 Predefined Hooks

If the variable bbdb-change-hook is set to the symbol bbdb-timestamp-hook (the default), then every record in the database will have a field named 'timestamp', which will always contain the date and time at which this record was created or last modified.

If the variable bbdb-create-hook is set to the symbol bbdb-creation-date-hook (the default), then every record in the database will have a field named 'creation-date', which will contain the date and time at which this record was added to the database.

If the variable bbdb/mail-auto-create-p is set to the symbol bbdb-ignore-most-messages-hook, then the variable bbdb-ignore-most-messages-alist will determine which messages should have records automatically created for them. The format of this alist is

```
(( HEADER-NAME . REGEXP ) ... )
```

for example,

```
(("From" . "@.*\\.maximegalon\\.edu")
 ("Subject" . "time travel"))
```

will cause **BBDB** entries to be made only for messages sent by people at Maximegalon U., or (that's *or*) people posting about time travel.

There may be only one entry per header in this alist: that is,

```
(("From" . "addr1\\|addr2") ... )
```

is legal, but

```
(("From" . "addr1") ("From" . "addr2") ... )
```

is not.

If the variable `bbdb/mail-auto-create-p` is set to the symbol `bbdb-ignore-some-messages-hook`, then the variable `bbdb-ignore-some-messages-alist` will determine which messages should have records automatically created for them. This is the exact inverse of the semantics of the `bbdb-ignore-most-messages-alist`: the alist specifies which messages should *not* have records automatically created for them, instead of which should. For example,

```
(("From" . "mailer.daemon")
 ("To" . "mailing-list-1\\|mailing-list-2")
 ("CC" . "mailing-list-1\\|mailing-list-2"))
```

will cause **BBDB** entries to not be made for messages from any mailer daemon, or messages sent to or **CC**ed to either of two mailing lists.

The variable `bbdb/news-auto-create-p` may be set to either of the above-mentioned functions as well, to get this behavior for netnews messages instead of mail messages.

If the variable `bbdb-notice-hook` is set to the symbol `bbdb-auto-notes-hook`, then the variable `bbdb-auto-notes-alist` may be used to automatically add text to the notes fields of the records corresponding to certain messages. The format of this alist is

```
(( HEADER-NAME
   (REGEXP . STRING) ... )
    ... )
```

for example,

```
(("To" ("-vm@" . "VM mailing list"))
 ("Subject" ("sprocket" . "mail about sprockets")
            ("you bonehead" . "called me a bonehead")))
```

will cause the text `"VM mailing list"` to be added to the notes field of the record corresponding to anyone you get mail from via one of the **VM** mailing lists. If, that is, `bbdb/mail-auto-create-p` is set such that the record would have been created, or if the record already existed.

The format of elements of this list may also be

```
(REGEXP FIELD-NAME STRING)
```

or

```
(REGEXP FIELD-NAME STRING REPLACE-P)
```

meaning add the given string to the named field. The field-name may not be '`name`', '`aka`', '`address`', '`phone`', or '`net`' (builtin fields) but must be either '`notes`', '`company`', or the name of a user-defined note-field.

```
("pattern" . "string to add")
```

is equivalent to

```
("pattern" notes "string to add")
```

STRING can contain `\&` or `\N` escapes like in the function `replace-match`. For example, to automatically add the contents of the **organization** field of a message to the `company` field of a **BBDB** record, you can use this:

```
("Organization" (".*" company "\\&"))
```

(Note you need two \ to get a single \ into a lisp string literal.)

If STRING is an integer *N*, the *N*th matching subexpression is used, so the above example could be written more efficiently as

```
("Organization" (".*" company 0))
```

If STRING is neither a string or an integer, it should be a function which is called with the contents of the field, and the result of the function call is used.

If the *REPLACE-P* flag is true, then the string replaces the old contents instead of being appended to it.

If multiple clauses match the message, all of the corresponding strings will be added.

If the string is being appended (*REPLACE-P* is false or not provided) then the new string is appended to the end of the existing field value, with an intervening newline. So each piece of text automatically added to this field will go on its own line.

You can control what the separator is by putting a `field-separator` property on the symbol naming the field. For example, to make text automatically added to a field named `newsgroups` be separated by commas, you could do

```
(put 'newsgroups 'field-separator "; ")
```

This variable works for news as well. You might want to arrange for this to have a different value when in mail as when in news.

There may be only one entry per header in this alist: that is,

```
(("Subject" ("\\bfoo\\b" . "Foo!!")
             ("bar" . "Bar!")))
```

will work, but

```
(("Subject" ("\\bfoo\\b" . "Foo!!"))
 ("Subject" ("bar" . "Bar!")))
```

will not.

Here's a more complicated example: some people include bitmaps of themselves in their mail messages in an **X-Face:** header field. You can capture this field into the '`*BBDB*`' with the following:

```
(setq bbdb-auto-notes-alist
      (append bbdb-auto-notes-alist
              (list "x-face"
                    (list (concat "[ \t\n]*\\([^ \t\n]*\\)"
                                  "\\([ \t\n]+\\([^ \t\n]+\\)\\)?"
                                  "\\([ \t\n]+\\([^ \t\n]+\\)\\)?"
                                  "\\([ \t\n]+\\([^ \t\n]+\\)\\)?"
                                  )
                          'face
                          "\\1\\3\\5\\7"))))
```

(The calls to `list` and `concat` are just for readability, it could easily be a constant.) The tricky bit here is that it strips out the newlines and whitespace used for header continuation, which are not actually a part of the face data. So though the mail message may have the face data on multiple lines, the entry in the '`*BBDB*`' will be just one line.

`bbdb-auto-notes-ignore` is an alist of headers and regexps to ignore in `bbdb-auto-notes-hook`. Each element looks like

(*HEADER* . *REGEXP*)

for example,

```
("Organization" . "^Gatewayed from\\|^Source only")
```

would exclude the phony `Organization:` headers in GNU mailing-lists gatewayed to the `gnu.*` newsgroups. Note that this exclusion applies only to a single field, not to the entire message. For that, use the variable `bbdb-auto-notes-ignore-all`.

`bbdb-auto-notes-ignore-all` is an alist of headers and regexps which cause the entire message to be ignored in `bbdb-auto-notes-hook`. Each element looks like

(*HEADER* . *REGEXP*)

for example,

```
("From" . "BLAT\\.COM")
```

would exclude any notes recording for message coming from `BLAT.COM`. Note that this is different from `bbdb-auto-notes-ignore`, which applies only to a particular header field, rather than the entire message.

1.8 Utilities

This section describes **BBDB** functionality that does not fit neatly into other sections.

1.8.1 bbdb-ftp

The `bbdb-ftp` utility enables the storage of FTP sites as **BBDB** records. The `bbdb-create-ftp-site` function is used to create a **BBDB** record for an FTP site. The command will prompt for information needed to create the record. The FTP site for a given record can be accessed with the `bbdb-ftp` command.

1.8.2 bbdb-print

`bbdb-print` is a utility for pretty-printing entries from the **BBDB** using TeX. It is invoked by pressing *P* in the `*BBDB*` buffer - this will cause all records in the **BBDB** to be printed as governed by the variables described below. If *P* is prefixed by a `*`, only the currently-displayed records will be printed. Once invoked, `bbdb-print` will generate the TeX output in a buffer named '`~/bbdb.tex`' (controlled by `bbdb-print-file-name`). The generated output will be shown, and must be **manually** saved.

For each record printed, `bbdb-print` will look for the presence of the `tex-name` field. If this field is found, its value will be printed instead of the `name` field for the record in question. This field is intended to allow the storage of names with accents or other characters that would be illegal in the address portion of a message. While other fields have special characters quoted by `bbdb-print` as described below, the contents of the `tex-name` field (if present) are used verbatim.

The following variables govern the printing of records (and of the printing of the fields therein):

`bbdb-print-elide`
: This variable should be set to a list of the fields that are not to be printed by `bbdb-print`. It defaults to:

```
(tex-name aka mail-alias)
```

`bbdb-print-file-name`
: The name of the file where generated TeX output is to be stored. The default is '`~/bbdb.tex`'.

`bbdb-print-require`
: The fields required for printing a record. This allows, for example, only records with phone numbers to be printed. The value of the value of the variable will be evaluated once for each record, and the record will be printed only if the evaluation returns a non-nil value. The symbols `name`, `company`, `net`, `phone`, `address`, and `notes` will be set to the appropriate values during evaluation; they will be nil if the field does not exist or is elided.

  The value of this variable can be any lisp expression, but typically it will be used for a boolean combination of the field variables, as in the following simple examples:

  - Print only people whose phone numbers are known:

```
(setq bbdb-print-require 'phone)
```

  - Print people whose names AND companies are known:

```
(setq bbdb-print-require '(and name company))
```

  - Print people whose names, and either addresses OR phone numbers are known:

```
(setq bbdb-print-require '(and name (or address phone))).
```

    The default value is:

```
(or address phone)
```

`bbdb-print-alist`
: Formatting options for all formats of `bbdb-print`. The value is an alist of the form

```
((OPTION . VALUE) (OPTION . VALUE) ...)
```

Separate settings can be configured using `bbdb-print-brief-alist` and `bbdb-print-full-alist`. Settings in these variables will override the ones in `bbdb-print-alist`.

The possible options and legal settings are:

`columns` `1`, `2`, `3`, `4`, `quad` (4 little 2-column pages per sheet) or `grid` (12 credit-card-sized pages per sheet).

`separator`
`0-7`, the style of heading for each letter. `0`=none, `1`=line, `2`=boxed letters, `3`=large boxed letters, `4`=large letters, `5`=letters with lines, `6`=letters with suits, `7`=boxed letters with suits.

`omit-area-code`
A regular expression matching area codes to omit.

`phone-on-first-line`
If `t`, the first phone number will be put on the same line as the name. If `nil`, the name will be the only text on the line. If set to a string, the field of type `phone` whose name matches the string will be used. The string should be a valid regular expression.

`n-phones` Maximum number of phone numbers to include.

`n-addresses`
Maximum number of addresses to include.

`include-files`
List of TeX files to `\input`. If the filenames are not absolute, the files must be located somewhere in the TeX input path.

`ps-fonts` Postscript fonts (TimesNewRoman and Courier) will be used if the value is non-`nil`. Standard TeX fonts (ec-fonts) will be used otherwise.

`font-size`
The integer point size of the font to be used.

`hsize` The horizontal dimension of the pages. The value must be a string, and must be a valid TeX dimension. Alternatively, the TeX default will be used if the value is `nil`.

`vsize` The vertical dimension of the pages. The value must be a string, and must be a valid TeX dimension. Alternatively, the TeX default will be used if the value is `nil`.

`hoffset` The TeX output will be offset to the right by the value of this option. The value must be either a string containing a valid TeX dimension or `nil` or `0` to use the default TeX horizontal offset.

`voffset` The TeX output will be offset downward by the value of this option. The value must be either a string containing a valid TeX dimension or `nil` or `0` to use the default TeX vertical offset.

`quad-hsize`
: Horizontal size to be used for the individual pages in the quad format. The value must be a string containing a valid TeX dimension.

`quad-vsize`
: Vertical size to be used for the individual pages in the quad format. The value must be a string containing a valid TeX dimension.

The default value is

```
((omit-area-code . "(AREA-CODE)")
 (phone-on-first-line . "^[ \t]*$")
 (ps-fonts . nil)
 (font-size . 6)
 (quad-hsize . "3.15in")
 (quad-vsize . "4.5in"))
```

Where *AREA-CODE* is the value of `bbdb-default-area-code` or '`000`' if there is no default area code.

`bbdb-print-brief-alist`
: Extra options for the `bbdb-print` brief format. The value(s) of this variable can either supplement or override the values in `bbdb-print-alist`. The format and possible values of this variable are as in `bbdb-print-alist`, described above.

The default value is

```
((columns . 1)
 (separator . 1)
 (n-phones . 2)
 (n-addresses . 1)
 (include-files "bbdb-print-brief" "bbdb-cols"))
```

`bbdb-print-full-alist`
: Extra options for the `bbdb-print` non-brief format. The value(s) of this variable can either supplement or override the values in `bbdb-print-alist`. The format and possible values of this variable are as in `bbdb-print-alist`, described above.

The default value is

```
((columns . 3)
 (separator . 2)
 (include-files "bbdb-print" "bbdb-cols"))
```

`bbdb-print-prolog`
: TeX statements to include at the beginning of the `bbdb-print` output file.

`bbdb-print-epilog`
: TeX statements to include at the end of the `bbdb-print` output file.

1.8.3 bbdb-snarf

`bbdb-snarf` provides the ability to generate **BBDB** records from raw text. If invoked as `bbdb-snarf`, it attempts to generate the record from the information around point (the cursor). If invoked as `bbdb-snarf-region`[6], the active region is used to generate the record.

Restrictions:

1. `bbdb-snarf` currently recognizes only US-style phone numbers.
2. `bbdb-snarf` works best with things that look like mailing addresses.

Example of an address that `bbdb-snarf` will recognize:

```
another test person
1234 Gridley St.
Los Angeles, CA 91342
555-1212
test@person.net
http://www.foo.bar/
other stuff about this person
```

1.8.4 bbdb-srv

`bbdb-srv` provides the ability to initiate the display of **BBDB** records from outside of Emacs. This allows external programs to cause the **BBDB** record for a given person to appear in the running Emacs when, for example, mail is recieved from that person. One specific application, described below, is the integration of Netscape and the **BBDB**, allowing the display of **BBDB** records corresponding to Mail and/or News messages displayed in Netscape.

`bbdb-srv` is composed of two parts - an external Perl script (‘`utils/bbdb-srv.pl`’), and an Emacs Lisp file (‘`lisp/bbdb-srv.el`’). The external portion is used to send commands to the internal portion.

NOTE: `bbdb-srv` requires `gnuserv` and `itimer`, both included with XEmacs. `gnuserv` must be started with the `gnuserv-start` command before `bbdb-srv` can be used.

In it’s most basic form, mail or news headers are passed to the Perl script. The Perl script then causes the **BBDB** record (if any) corresponding to the passed headers to be displayed in the running Emacs. While `bbdb-srv` will operate with just a ‘`From:`’ field, it works better when passed both ‘`From:`’ and ‘`To:`’ headers. When `bbdb-srv` notices that the logged-in user is named in the ‘`From:`’ header, it will attempt to display the record (if any) for the person named in the ‘`To:`’ header. If no ‘`To:`’ header is passed, it falls back on the record (if any) for the logged-in user (the person named in the ‘`From:`’ header).

An example manual invocation of `bbdb-srv` is as follows:

```
% cat |bbdb-srv.pl
From: Jamie Zawinski <jwz@netscape.com>
To: Matt Simmons <simmonmt@acm.org>
⟨CTRL-D⟩
```

[6] `bbdb-snarf` is actually a wrapper for `bbdb-snarf-region` that determines the relevant region and passes it on.

If the invoking user is Jamie Zawinski, the record for Matt Simmons (if any) will be displayed. If the invoking user is not Jamie Zawinski, the record for Jamie Zawinski (if any) will be displayed.

As mentioned above `bbdb-srv` can be used with Netscape Mail and Netscape News. Please note that it can only be used with the UNIX versions 3.0b2 and greater of these applications. To allow Netscape to use `bbdb-srv`, set the '`NS_MSG_DISPLAY_HOOK`' variable to `bbdb-srv.pl`[7] as follows:

Bourne Shell ('`/bin/sh`') and variants:

```
# NS_MSG_DISPLAY_HOOK bbdb-srv.pl
# export NS_MSG_DISPLAY_HOOK
```

C-Shell ('`/bin/csh`') and variants:

```
% setenv NS_MSG_DISPLAY_HOOK bbdb-srv.pl
```

The following variables can be used to customize the behavior of `bbdb-srv`:

`bbdb/srv-auto-create-p`

This variable is similar to `bbdb/news-auto-create-p` and `bbdb/mail-auto-create-p`. That is, when headers are passed in to `bbdb-srv`, a new **BBDB** record can be created if none exists, depending on the value of this variable. Possible values are:

`t` Automatically create new **BBDB** records if the headers passed in do not correspond to an already-existing record.

`nil` Do not automatically create new **BBDB** records.

FUNCTION

FUNCTION is called. If it returns `t`, a record will be created for the person named in the '`From:`' header. If it returns `nil`, no record will be created.

A suggested function for use is `bbdb/srv-auto-create-mail-news-dispatcher`. This function will attempt to determine the source of the passed headers - whether they were part of a mail message or of a news article. The action (if any) dictated by the value of either `bbdb/mail-auto-create-p` or `bbdb/news-auto-create-p`, based on the determined source of the passed headers.

`bbdb/srv/display-delay`

`bbdb-srv` pauses between displaying the records corresponding to each passed set of headers. This variable controls the length of time (in seconds) of the delay between the display of different records. Note when setting this variable that only one set of headers can be queued at a time. If three sets of headers are passed to `bbdb-srv` in less than the delay time, only the first and last will be displayed.

[7] Use the full path to `bbdb-srv.pl` if it is not in the default path.

1.9 Internals

This section is currently a dumping ground for things that should eventually go here, but were found elsewhere in the file.

INFORMATION IN THIS SECTION IS FOR INFORMATIONAL PURPOSES ONLY. IT SHOULD NOT BE TAKEN AS DOCUMENTATION OF AN EXTERNAL API. EVERYTHING LISTED BELOW IS SUBJECT TO CHANGE WITHOUT NOTICE

The first time you use one of the **BBDB** commands, this file is read into an emacs buffer, and remains there. As you make changes to the database, this buffer is changed as well, ensuring that if it is auto-saved, it will be saved in its most current state.

1.9.1 BBDB data file format

The data file is arranged in a hierarchical fashion. At the top level are vectors, with one vector per database record. It is **very** important that each vector be on its own line, as the BBDB builds and stores markers based on this layout. The markers are then used to increase the speed of database modifications (more on this later). The record vectors contain the individual fields of the record. These fields can be of any type, but are currently integers, strings, lists of strings, alists, vectors, or lists of vectors. In the case of fields that contain one or more vectors, they can be further broken down in terms of the fields of their component vectors.

In an effort to provide a more concrete example to illustrate the above, and to provide a reference for database accessor and modifier functions, we describe the database format below. This description starts with the fields of the individual record vectors, and drills down through the vectors used by some of the fields.

1.9.1.1 Record Vectors

Name	Type	Accessor and Modifier	Description
First name	String	`bbdb-record-firstname` `bbdb-record-set-firstname`	Entity's first name
Last name	String	`bbdb-record-lastname` `bbdb-record-set-lastname`	Entity's last name
AKAs	List of Strings	`bbdb-record-aka` `bbdb-record-set-aka`	Alternate names for entity
Company	String	`bbdb-record-company` `bbdb-record-set-company`	Company with which entity is associated
Phones	List of Vectors	`bbdb-record-phones` `bbdb-record-set-phones`	List of phone number vectors
Addresses	List of Vectors	`bbdb-record-addresses` `bbdb-record-set-addresses`	List of address vectors
Net address	List of Strings	`bbdb-record-net` `bbdb-record-set-net`	List of network addresses

Notes	String or Alist	`bbdb-record-raw-notes` `bbdb-record-set-raw-notes`	String or Association list of note fields (strings)
Cache	Vector	`bbdb-record-cache` `bbdb-record-set-cache`	Record cache. *Internal version only.*

The phone, address and cache vector fields are described below. Please note that, as indicated in the table above, the cache is present only in the internal version of the database - it is not written out as part of the '`.bbdb`' file.

In addition, the accessor and modifier functions for the notes alist are described.

1.9.1.2 Phone Vectors

To access the fields in the below table, you must first get the list of phone vectors using the `bbdb-record-phones` function. Note that if you alter the phones field with the `bbdb-record-set-phones` function, you are altering the entire phones list for the given record. Use the modifier functions below for modifications to individual phone vectors.

Name	Type	Accessor and Modifier	Description
Location	String	`bbdb-phone-location` `bbdb-phone-set-location`	Phone number identifier
Area	Integer	`bbdb-phone-area` `bbdb-phone-set-area`	Area code for phone number
Exchange	Integer	`bbdb-phone-exchange` `bbdb-phone-set-exchange`	Exchange (aka prefix) for phone number
Suffix	Integer	`bbdb-phone-suffix` `bbdb-phone-set-suffix`	Suffix for phone number
Extension	Integer	`bbdb-phone-extension` `bbdb-phone-set-extension`	Phone number extension ('`0`' if none)

1.9.1.3 Address Vectors

To access the fields in the below table, you must first get the list of address vectors using the `bbdb-record-addresses` function. Note that if you alter the addresses field with the `bbdb-record-set-addresses` function, you are altering the entire addresses list for the given record. Use the modifier functions below for modifications to individual address vectors.

Name	Type	Accessor and Modifier	Description
Location	String	`bbdb-address-location` `bbdb-address-set-location`	Address identifier
Streets	List	`bbdb-address-streets` `bbdb-address-set-streets`	List of street address lines. `nil` if none.

Street3	String	`bbdb-address-street3` `bbdb-address-set-street3`	Third line of street address. "" if none.
City	String	`bbdb-address-city` `bbdb-address-set-city`	City name
State	String	`bbdb-address-state` `bbdb-address-set-state`	State abbreviation
Zip	Integer (american) List (non-american)	`bbdb-address-zip` `bbdb-address-set-zip`	Zip code
Country	String	`bbdb-address-country` `bbdb-address-set-country`	Country

1.9.1.4 Cache Vector

This vector is present only in the internal database representation. It is not written out to the database file because it contains information aggregated from the rest of the record that is reconstructed when the database is read. To write the cache information to the database file would increase the risk of database inconsistency, and would violate the principles of normalization.

To access the cache fields using the functions listed below that begin with `bbdb-cache-`, you must first get the cache vector using the `bbdb-record-cache` function. The functions that begin with `bbdb-record-` get the cache vector internally. Note that if you alter the cache field in the high-level record with the `bbdb-record-set-cache` function, you are altering the entire cache vector for the given record. Use the modifier functions below for modifications to individual cache fields.

Name	**Type**	**Accessor and Modifier**	**Description**
Name Cache	String	`bbdb-cache-namecache` `bbdb-cache-set-namecache`	Preconcatenated name of entity
Sort Key	String	`bbdb-cache-sortkey` `bbdb-cache-set-sortkey`	Preconcatenated sort key for record

Marker	Marker	`bbdb-cache-marker`
		`bbdb-record-marker`
		`bbdb-cache-set-marker`
		`bbdb-record-set-marker`
			Marker in '`.bbdb`' for start of record

Deleted	Boolean	`bbdb-cache-deleted-p`
		`bbdb-record-deleted-p`
		`bbdb-cache-set-deleted-p`
		`bbdb-record-set-deleted-p`
			Set to `t` if record has been deleted, `nil` if not

The functions listed above will return `nil` if their respective cache fields are not set. The functions listed below will return the value of their cache fields if set, but will also build (and set) the correct field values if the fields are unset:

`bbdb-record-name`
: Return the name in the Name Cache field of the cache (if set). If the name has not been built yet (if the field is `nil`), the name is built, stored in the Name Cache field, and returned.

`bbdb-record-sortkey`
: Return the name it the Sort Key field of the cache (if set). If the Sort Key field has not yet been set (if the field is `nil`), the Sort Key is built, stored in the Sort Key field, and returned.

1.9.1.5 Notes String or Alist

If there is only a single note for a given record, the notes field for that record will be a string. If there is more than one note, the notes field will be an association list (alist) with elements of the form

(*NAME* . *VALUE*)

where *NAME* is the symbol for the name of the note, and *VALUE* is the value of the note.

1.9.1.6 Example BBDB record

1.10 Mailing Lists

(If you are looking for a way to create mailing lists with **BBDB**, you should be looking at the section on See Section 1.4.3 [Mail Sending Interfaces], page 18.)

There are three mailing lists for the **BBDB**. `bbdb-info@lists.sourceforge.net` gets moderate traffic, and is intended for the discussion and distribution of development versions of the **BBDB**. Users of development versions of the **BBDB** should be subscribed to this list. It is also to this list that bugs should be reported. See Section 1.13.1 [Known Bugs], page 43. for instructions on submitting bug reports.

The second mailing list `bbdb-announce@lists.sourceforge.net` has very low volume. Any user of the **BBDB** should consider subscribing to this list, as new releases and security issues will be posted here.

A third mailing list, `bbdb-cvs@lists.sourceforge.net`, is intended for developers to follow the changes made to the **BBDB** development version. Developers of the **BBDB** should consider to subscribe to this list.

1.11 Changes in this Version

1.11.1 Major Changes

Database File Version Change

(the following version-migration text will move in a future version. It is in this section currently because new users will likely not experience it)

There has been a version change in the **BBDB** database file. The new version supports non-US zip codes, and an additional "Country" field in addresses.

An automatic version-migration mechanism has been implemented that allows older version '`.bbdb`' files to either be migrated to the new version, or used as-is without migrating.

When the **BBDB** detects a database file with an old version, it will display the features that have been introduced **in the database file** from the time of the older version's implementation. It will offer the choice of migration or use of the **BBDB** with the older-version file.

If migration is chosen, the database file will be automatically changed to the new format. If migration is declined, the file will kept in the older format in the `.bbdb` buffer, but will be stored internally in the new format. When changes need to be made to the `.bbdb` buffer, changed records will be reverse-migrated from their internal version to that of the disk file.

1.11.2 Other Changes

TeX Output

By default, ec fonts are used for TeX output instead of cm fonts. With the `ps-fonts` option set in `bbdb-print-alist`, TimesNewRoman and Courier fonts are used.

1.12 The Latest Version

Released versions of the **BBDB** can be found at the following site:

- WWW: `http://bbdb.sourceforge.net`
- FTP: `ftp://ftp.sourceforge.net/pub/bbdb`

Development versions of the **BBDB** can be obtained in the following ways:

- WWW: `http://bbdb.sourceforge.net`
- Anonymous CVS: See `http://bbdb.sourceforge.net` for instructions.

Users of development versions of the **BBDB** should subscribe to the `bbdb-info` mailing list. See Section 1.10 [Mailing Lists], page 42.

1.13 The Future

The future consists of Bugs and Features.

1.13.1 Known Bugs

1. ***M-TAB* conflicts with ispell.** Workaround: The binding installed by the **BBDB** for address completion/expansion conflicts with that used by ispell. The suggested workarounds are to rebind the ispell key (the **BBDB** binding is not configurable at this time), to manually invoke ispell via *M-x*, or to not use ispell completion functionality in **BBDB**-enabled message composition buffers. The following is an example of such a rebinding, supplied by Kai Grojohan:

   ```
   (defun my-message-mode-keys ()
     (define-key message-mode-map (kbd "M-TAB") 'bbdb-complete-name))
     (add-hook 'message-mode-hook 'my-message-mode-keys)
   ```

2. **BBDB and abbrev expansion is inconsistent.** Workaround: Currently, *M-TAB* must be used to expand/complete against **BBDB** names and net addresses, and *TAB* must be used to expand abbrevs (the values in the `mail-alias` field). Unification is planned for a future version.

3. **The `*BBDB*` buffer does not always come up when the first article in a Gnus Summary Buffer is selected.** Workaround: Pressing *g* to reload the article. This will cause the `*BBDB*` buffer to be displayed.
4. **Expansion will fail when the name to be expanded is a subset of the name for another record.** For example, if you have entries for '`John`' and '`Johnathan`', you will not be able to expand the name for '`John`'. Workaround: Use the net address for the subset name ('`John`' in this example).

It is commonly known that there are no bugs in the **BBDB**. Bugs found in defiance of this rule should be submitted to the `bbdb-info` mailing list (see Section 1.10 [Mailing Lists], page 42). To assist the developers, please include the version numbers of the various programs used when the bug occurred. An example report follows.

```
BBDB Version:
Emacs/XEmacs Version:
Mail/News reader (Gnus, VM, etc) used:
Bug Description:
```

1.13.2 TODO List

The Near Future

- Add bbdb/*MUA*-delete-sender-record
- Configurable completion. Should allow user to specify "complete on names first, then nets", etc.
- More variables for upgrading. Specifically a variable that lets users specify extra fields for upgrading (an alist '`(a . b)`' that says field '`a`' should be upgraded the same way as field '`b`'.
- Change all functions that switch on MUAs to use compose-mail (ex: `bbdb-send-mail-internal`).
- Soren Dayton's method for generically extending the **BBDB** with special-purpose fields
- Sorting records on alternate keys. `bbdb-sort-by` from Boris Goldowsky.
- Sorting individual types of fields - Sam Steingold's method.
- Button 3 menus (Mark Moll and Soren Dayton)
- Conditionalized erasure of properties when text is pasted into the **BBDB**.
- Should notice when there are '`Reply-To:`' addresses.
- Should have a command for merging together two divergent copies of a '`.bbdb`' file (in case you read mail on one machine and news on another, for instance.)
- The '`*BBDB*`' buffer should be resized to exactly fit what it's displaying, even when not in "electric" mode.
- It should be possible to do completion on last names as well as first names.
- The BBDB buffer is left at the top of the stack when GNUS is exited because GNUS runs its exit-hooks too early. This should be fixed.
- String area codes (German area codes can begin with zeroes) patch from `Michael Sperber <sperber@informatik.uni-tuebingen.de>`
- Default country variable, similar to `bbdb-default-area-code`.

- Make format self-describing in comment
- ISO-8859-x characters in records for printing.
- Prefix for `W` (`bbdb-www`) command to allow selection of different addresses.
- Generalized buttons (via extents) for fields. Example: '`(a . b)`' means create button that calls '`b`' for each entry in the '`a`' field.
- Remove support for GNUS. Start with lisp Makefile (remove nntp and gnus loads).
- More flexible auto-addition. Conditionalizing of addition (conditions or supplied function). Prompt if multiple records that meet criteria exist.
- Different output formats. See Toby Speight's `<s8iurdodvm.fsf@plato.ansa.co.uk>` and Bin Mu's `<199801221605.KAA23663@DerivaTech.Com>`.
- Generalized area-code-split program that could split, for example, based on input copied (or straight fetch of page) from the Bellcore NANP page.
- Print multivalue (comma-separated) fields with one value per line
- Easier BBDB extension. See `<xcdyb30f3hb.fsf@ra.cs.uchicago.edu>` from Soren Dayton.
- Take birthdays from the **BBDB**, add them to calendar. From Boris Goldowsky.
- Make mail aliases file for other mailers. From Boris Goldowsky.
- Various patches from Boris Goldowsky in '`bbdb-ext`'.
- Various other patches:
  - `bbdb-filters-0.2`
  - `bbdb-frame.el`
  - `bbdb-letter-1.0`
  - `bbdb-plz`
  - `bbdb-query`
  - `country`
  - `country-info`

  Note that these files have not been investigated. They may or may not be incorporated.
- Ability to remove all properties from copied strings.

Not-So-Near Future

- Fix Gnus scoring so it rebuilds when gnus-score disappears
- Multiline note fields
- Change key to be some kind of unique number
- There should be better support for non-American addresses and phone numbers. This might be Near Future if somebody volunteers to send me patches.
- Should reimplement "electric" mode to not be so broken.
- The `*C-o` keystroke should add a field to all displayed records. Perhaps `*;` should append some text to an arbitrary field of all displayed records.
- Multiple '`.bbdb`' files with precedence relationships. See Wes Hardaker's `<sdu3awz75a.fsf@oakdale.ucdavis.edu>`

- Automatically grab information about a person from their sig. See Graham Clark's `info-bbdb` post `<6282.199706161624@havra.dcs.ed.ac.uk>` and Adrian Aichner's `info-bbdb` post `<rxsiuzebpgp.fsf@midnight.ecf.teradyne.com>`. Would like to have `bbdb-snarf` attack the sig then compare the snarfed data with the header data.

Thoughts

- Are there enough hooks?
- The interfaces should share more code.
- The `bbdb-create-internal` function should be more forgiving.
- More `*` commands in general, including `*d`.

1.13.3 End of Life (EOL) Statements

The items in the following list describe items for which support will be removed in coming versions of the **BBDB**. The items listed are guaranteed to be supported and present only until the EOL date. They may be removed without warning at any time thereafter.

1. `advertized-bbdb-delete-current-field-or-record`
   Support for this function will be removed for version 2.2. It is recommended that all code depending on this variable be switched to use `bbdb-delete-current-field-or-record`. The two functions have the same calling conventions and effects. This EOL statement was added for version 2.1.
2. Support for the GNUS (not Gnus) newsreader
   The GNUS-specific parts of the **BBDB** will be actively removed for the 2.2 release. No further maintenance and/or bugfixes are planned for GNUS code at this time. This EOL statement was added for version 2.1.

1.14 Thanks

Thanks to everyone on the info-bbdb mailing list for many useful suggestions. This hack would be far less insidious without their input!

Thanks list for versions after 2.00.06.

Thanks to Alex Schroeder, Ronan Waide, Thomas DeWeese, Robert Fenk, Didier Verna, Bill Carpenter.

Thanks list for versions after 1.51 prior to and including 2.00.06.

Thanks to Adrian Aichner, Kees de Bruin, David Carlton, Soren Dayton, Brian Edmonds, Boris Goldowsky, Seth Golub, John Heidemann, Christopher Kline, Carsten Leonhardt, Hrvoje Niksic, Jens-Ulrik Hoger Petersen, Colin Rafferty, Matt Simmons, Sam Steingold, Marco Walther, Christoph Wedler.

Thanks list for versions prior to and including 1.51.

And special thanks to Sebastian Kremer, Joe Wells, Todd Kaufmann, Andy Norman, Ivan Vazquez, Stewart Clamen, Roland McGrath, Dave Brennan, Kimball Collins, Dirk Grunwald, Philippe Queinnec, Boris Putanec, Dave Disser, Francois Felix Ingrand, Sean Owens, Guido Bosch, Lance Brown, Tom Emerson, George Hartzell, Luis Miguel Silveira, Kimmo Suominen, Derek Upham, David Zuhn, Rod Whitby, Richard Mlynarik.

Last, but not least, thanks to Jamie Zawinski for writing **BBDB** in the first place.

Concept Index

%

%F 22
%UB 22

*

BBDB buffer 13

A

Adding new fields 13
Adding new records 13
Address completion 18, 27
Address display 25
AKA 26
Alternate names 26
Annotating the sender of the current message . . 17
Auto-save files 26
Automatic display of the corresponding record 26
Automatically adding text to records 31
Automatically creating records 25, 30

B

BBDB Mode 13
Browsing the Web page for the current record . . 16
Builtin field types 9

C

Changing fields 13
Changing windows 16
Checking zip codes 24
Completion 27
Continental addresses 25
Creating new records 13, 16

D

Deleting fields 13
Dialling phone numbers 14
Display of addresses 25
Displaying changed records 16

E

Editing fields 13
Electric display 25
Elided display 12
European addresses 25
European phone numbers 13, 24

F

Finger interface 15
Fixing mistakes 14
Format of completed address 18
Formatting addresses 25

G

GNUS Subject-buffer 22

I

Initial Configuration 5
Inserting new fields 13
Installation 1
Invalid zip codes 24

L

List of valid zip codes 24

M

Mail address completion 18, 27
Mail Aliases 18
Mailing lists 18
Makefile 1
mark-char 22
Merging records 14

N

Name completion 18, 27
Normal User Installation 2
North American phone numbers 13, 24
Not a valid zip code 24

O

One-line display 12

P

Phone numbers 13, 24
Printing records in the database 11, 16, 33

R

Removing fields 13
Reordering fields 13

S

Saving your changes 14
Searching the database 12, 16
Sending mail 14, 18
Showing the sender of the current message 17
Special Field Names 10
Storing FTP sites in the BBDB 33

T

Timestamping records 30
Transposing fields 13

U

User-defined fields 12, 13

V

Valid zip codes 24

X

XEmacs Package Installation 4

Z

Zip code checking 24

Variable Index

B

bbdb-address-formatting-alist 25
bbdb-after-read-db-hook 29
bbdb-always-add-addresses 27
bbdb-auto-notes-alist 31
bbdb-auto-notes-ignore 33
bbdb-auto-notes-ignore-all 33
bbdb-auto-revert-p 26
bbdb-canonicalize-net-hook 29
bbdb-canonicalize-redundant-nets-p 28
bbdb-case-fold-search 25
bbdb-change-hook 28, 30
bbdb-check-zip-codes-p 24
bbdb-complete-name-allow-cycling 27
bbdb-complete-name-full-completion 27
bbdb-completion-display-record 18
bbdb-completion-type 27
bbdb-continental-zip-regexp 25
bbdb-create-hook 28
bbdb-default-area-code 24
bbdb-dial-local-prefix 15
bbdb-dial-local-prefix-alist 15
bbdb-dial-long-distance-prefix 15
bbdb-dwim-net-address-allow-redundancy ... 18
bbdb-electric-p 25
bbdb-expand-mail-aliases 27
bbdb-file 24
bbdb-finger-host-field 11, 15
bbdb-get-addresses-from-headers 19
bbdb-get-addresses-headers 19
bbdb-get-addresses-to-headers 19
bbdb-get-only-first-address-p 19
bbdb-ignore-most-messages-alist 30
bbdb-ignore-some-messages-alist 31
bbdb-info-file 16
bbdb-initialize-hook 29
bbdb-legal-zip-codes 24
bbdb-list-hook 28
bbdb-load-hook 29
bbdb-message-caching-enabled 28
bbdb-mode-hook 29
bbdb-new-nets-always-primary 27
bbdb-north-american-phone-numbers-p 24
bbdb-notice-auto-save-file 26
bbdb-notice-hook 29, 31
bbdb-offer-save 28
bbdb-pop-up-target-lines 27
bbdb-print-alist 34
bbdb-print-brief-alist 36
bbdb-print-elide 34
bbdb-print-epilog 36
bbdb-print-file-name 34
bbdb-print-full-alist 36
bbdb-print-prolog 36
bbdb-print-require 34
bbdb-quiet-about-name-mismatches 26
bbdb-readonly-p 26
bbdb-send-mail-style 14
bbdb-use-alternate-names 26
bbdb-use-pop-up 26
bbdb-user-mail-names 27
bbdb/gnus-header-prefer-real-names 22
bbdb/gnus-header-show-bbdb-names 22
bbdb/gnus-lines-and-from-length 22
bbdb/gnus-mark-known-posters 22
bbdb/gnus-score-default 19
bbdb/gnus-score-field 11, 19
bbdb/mail-auto-create-p 25, 31
bbdb/news-auto-create-p 25, 31
bbdb/sc-attribution-field 11
bbdb/srv-auto-create-p 38
bbdb/srv-display-delay 38
bbdb/vm-auto-add-label-field 22
bbdb/vm-auto-add-label-list 22
bbdb/vm-set-auto-folder-alist-field 22

G

gnus-score-find-score-files-function 19

V

vm-summary-format 22
vm-summary-uninteresting-senders 22

Table of Contents

1 BBDB 1

1.1 Installation 1
1.1.1 General Prerequisites 1
1.1.2 Normal User Installation 2
1.1.3 XEmacs Package Installation 4
1.1.4 Initial Configuration 5
1.1.4.1 Initializing **BBDB** support for Gnus 7
1.1.4.2 Initializing **BBDB** support for MH-E 7
1.1.4.3 Initializing **BBDB** support for RMAIL 7
1.1.4.4 Initializing **BBDB** support for Sendmail 7
1.1.4.5 Initializing **BBDB** support for VM 8
1.1.4.6 Initializing **BBDB** support for Message mode 8
1.1.4.7 Initializing **BBDB** support for Reportmail 8
1.1.4.8 Initializing **BBDB** support for Supercite 8
1.1.4.9 Initializing **BBDB** support for Web Browsers 9
1.2 The **BBDB** 9
1.2.1 Database Fields 9
1.2.2 Basic searching commands 11
1.2.3 Advanced searching commands 12
1.2.4 Manual record addition 12
1.3 BBDB Mode 13
1.3.1 Functions bound to keys in **BBDB** Mode 13
1.3.2 Other database manipulation functions 16
1.4 Interfaces 17
1.4.1 Mail Reading Interfaces 17
1.4.2 News Reading Interfaces 17
1.4.3 Mail Sending Interfaces 18
1.5 Reader-specific Features 19
1.5.1 Gnus-specific Features 19
1.5.1.1 Scoring 19
1.5.1.2 Gnus Summary Buffer Enhancements 20
1.5.1.3 GNUS Summary Buffer Enhancements 21
1.5.2 VM-specific features 22
1.5.2.1 VM Message Summary Enhancements 22
1.5.2.2 VM configuration of what records the **BBDB** buffer shows 22
1.5.2.3 VM automatic setup of `vm-set-auto-folder-alist` 22
1.5.2.4 VM automatic adding of labels 22
1.6 Using the **BBDB** with other packages 23
1.6.1 Using the **BBDB** with Message Mode 23
1.6.2 Using the **BBDB** with Reportmail 23
1.6.3 Using the **BBDB** with Supercite 23

1.6.4 Using the **BBDB** with Web Browsers . . . 24
1.7 Options . . . 24
1.7.1 Customization Parameters . . . 24
1.7.2 Customization Hooks . . . 28
1.7.3 Predefined Hooks . . . 30
1.8 Utilities . . . 33
1.8.1 `bbdb-ftp` . . . 33
1.8.2 `bbdb-print` . . . 33
1.8.3 `bbdb-snarf` . . . 36
1.8.4 `bbdb-srv` . . . 37
1.9 Internals . . . 39
1.9.1 BBDB data file format . . . 39
1.9.1.1 Record Vectors . . . 39
1.9.1.2 Phone Vectors . . . 40
1.9.1.3 Address Vectors . . . 40
1.9.1.4 Cache Vector . . . 41
1.9.1.5 Notes String or Alist . . . 42
1.9.1.6 Example BBDB record . . . 42
1.10 Mailing Lists . . . 42
1.11 Changes in this Version . . . 42
1.11.1 Major Changes . . . 42
1.11.2 Other Changes . . . 43
1.12 The Latest Version . . . 43
1.13 The Future . . . 43
1.13.1 Known Bugs . . . 43
1.13.2 TODO List . . . 44
1.13.3 End of Life (EOL) Statements . . . 46
1.14 Thanks . . . 46

Concept Index . . . 48

Variable Index . . . 50